VERIFICATION OF TRANSLATION

I, Sayaka KOSUGE,

Of 4th Floor, MATRICE 2-13-7, MINAMIAOYAMA MINATO-KU,
TOKYO, 107-0062, JAPAN,

am the translator of the documents attached and I state that the following is a true translation to the best of my knowledge and belief of Japanese Patent Application No. 2003-165859.

Dated this on July 6, 2009

Signature of translator

Sayaka Kosuge

[Name of the Document] Request
[Reference No.] 0390470804
[Date of Filing] June 11, 2003
[Address] Commissioner of Patent Office

[Inventor]
[Address] c/o Sony Corporation
7-35, Kita-shinagawa 6-chome, Shinagawa-ku,
Tokyo
[Name] Takeshi Oka

[Inventor]
[Address] c/o Sony Corporation
7-35, Kita-shinagawa 6-chome, Shinagawa-ku,
Tokyo
[Name] Satoshi Katsuo

[Inventor]
[Address] c/o Sony Corporation
7-35, Kita-shinagawa 6-chome, Shinagawa-ku,
Tokyo
[Name] Takashi Furukawa

[Applicant]
[ID No.] 000002185
[Name] c/o Sony Corporation

[Agent]
[ID No.] 100082131
[Patent Attorney]
[Name] Yoshio Inamoto
[Telephone No.] 03-3369-6479

[Indication of Charge]
[Ledger Account No.] 032089
[Amount of Payment] 21,000 yen

[List of Document Filed]

[Document]	Claims	1
[Document]	Specification	1
[Document]	Drawings	1
[Document]	Abstract	1
[General Power of Attorney ID No.]	9708842	
[Necessity of Proof]		Needed

[NAME OF DOCUMENT] SPECIFICATION

[TITLE OF THE INVENTION] FILE GENERATION APPARATUS, METHOD, PROGRAM, AND RECORDING MEDIUM

[WHAT IS CLAIMED IS:]

[CLAIM 1] A file generation apparatus for generating a file of first data to be recorded on a recording medium, characterized by comprising:

a first data generation means for generating second data to be arranged at the beginning of the file;

a second data generation means for generating third data to be arranged at the end of the file; and

a third data generation means for generating fourth data which allows the data amount of the first, second, or third data to be an integral multiple of a unit of reading or writing to the recording medium by adding the fourth data to the first data, the second data, or the third data.

[CLAIM 2] The file generation apparatus according to claim 1, characterized in that

the first data generation means generates the second data, i.e., a header of the file.

[CLAIM 3] The file generation apparatus according to claim 1, characterized in that

the first data generation means further includes a format conversion means for converting the first data into a KLV (Key, Length, Value) structure; and

the first data generation means generates the second data composed of the file's header, and a key and a length arranged between the header and the first data.

[CLAIM 4] The file generation apparatus according to claim 1, characterized in that

the third data generation means generates the fourth data which allows the data amount of the first data divided into N-1 portions to be an integral multiple of a physical unit area of the recording medium, and the total data amount of the first data to be an integral multiple of the unit of reading and writing on the recording medium by adding the fourth data to each of the N-1 portions of the first data toward the beginning out of the first data divided into N portions, where N is an integer.

[CLAIM 5] The file generation apparatus according to claim 1, characterized in that

the third data generation means generates the fourth data for the first data divided into units corresponding to specified reproduction times with video data and audio data for a plurality of channels multiplexed in accordance with the divided units, the fourth data allows the data amount for each of divided units of the first data to be an integral multiple of the unit of reading and writing on the recording medium.

[CLAIM 6] The file generation apparatus according to claim 5, characterized in that

the third data generation means generates the fourth data which allows the data amount totaling partition data for separating divided portions of the first data from each other, metadata contained in each of divided portions of the first data, and the video data to be an integral multiple of the unit of reading and writing on the recording medium.

[CLAIM 7] The file generation apparatus according to claim 5, characterized in that

the third data generation means generates the fourth data which allows the data amount of each of divided portions of the audio data contained in each of divided portions of the first data to be an integral fraction of the unit of reading and writing on the recording medium, and the total data amount of the audio data to be an integral multiple of the unit of reading and writing on the recording medium.

[CLAIM 8] A method of generating a file of first data recorded on a recording medium, characterized by comprising:

a first data generation step of generating second data to be arranged at the beginning of the file;

a second data generation step of generating third data to be arranged at the end of the file; and

a third data generation step of generating fourth data which allows the data amount of the first, second, or third data to be an integral multiple of a unit of reading or writing to the recording medium by adding the fourth data to the first data, the second data, or the third data.

[CLAIM 9] A program which causes a computer to perform a file generation process of generating a file of first data, characterized by comprising:

a first data generation step of generating second data to be arranged at the beginning of the file;

a second data generation step of generating third data to be arranged at the end of the file; and

a third data generation step of generating fourth data which allows the data amount of the first, second, or third data to be an integral multiple of a unit of reading or writing to the recording medium by adding the fourth data to the first data, the second data, or the third data.

[CLAIM 10] A recording medium to record a file of first data, characterized in that

the recording medium records first data whose data amount is made to correspond to an integral multiple of a unit of reading or writing to the recording medium by adding first additional data to the first data, so that a boundary of the first data matches a boundary of the unit;

the recording medium records second data to be arranged at the beginning of the file whose data amount is made to correspond to an integral multiple of the unit by adding second additional data to the second data, so that a boundary of the second data matches a boundary of the unit; and

the recording medium records third data to be arranged at the end of the file whose data amount is made to correspond to an integral multiple of the unit by adding third additional data to the third data, so that a boundary of the third data matches a boundary of the unit.

[DETAILED DESCRIPTION OF THE INVENTION]

[FIELD OF THE INVENTION]

[0001]

The present invention relates to a file generation apparatus, method, program, and recording medium. More specifically, the present invention relates to a file generation apparatus, method, program, and recording medium for generating or recording files suitable for recording on a recording medium..

[BACKGROUND ART]

[0002]

In recent years, with standardizing of communication protocols and the like and decreasing of prices of communication devices and the like, personal computers that have communication I/Fs (Interfaces) as standard equipment have become general.

[0003]

Further, in addition to the personal computers, professional-use broadcast equipments such as AV (Audio Visual) servers and VTRs (Video Tape Recorders) having or being able to have the communication I/Fs as standard equipment have also been general. The broadcast equipments exchange files of video data and audio data (hereafter collectively referred to as AV data) with each other.

[0004]

Incidentally, in the past, formats specific to for example models and manufacturers are used as formats for files to be exchanged between broadcast equipments. As a result, exchanging files between broadcast equipments of different models or manufacturers has been difficult.

[0005]

In this regard, for example, MXF (Material eXchange Format) is proposed as a file exchange format and is currently in the process of standardization.

[DISCLOSURE OF THE INVENTION]

[PROBLEMS TO BE SOLVED BY THE INVENTION]

[0006]

MXF is a format in consideration for streaming in addition to file exchange and multiplexes video data and audio data in fine units such as frames.

[0007]

As mentioned above, MXF multiplexes video data and audio data for each frame in consideration for streaming. Accordingly, it has been difficult to incorporate an MXF file into a storage and then separately edit video data and audio data (AV independent editing).

[0008]

In this regard, there is a method of incorporating, in the broadcast equipment, an MXF file and then converting it into a specifically formatted file. However, when the broadcast equipment converts an MXF file into a file formatted completely irrelevantly to MXF and records it on the storage, it becomes difficult to handle that file in another broadcast equipment.

[0009]

In other words, for example, in a case where a given broadcast equipment records a specifically formatted file on a storage thereof and another broadcast equipment accesses that file via a communication I/F such as IEEE (Institute of Electrical and Electronics Engineers) 1394 or USB (Universal Serial Bus), when the other broadcast equipment cannot comprehend that specific format, this broadcast equipment cannot handle (or read, in this example) the specifically formatted file.

[0010]

Further, in a case where a given broadcast equipment records a specifically formatted file on a removable recording medium such as an optical disc, when the removable recording medium is mounted on another broadcast equipment and the other broadcast equipment cannot comprehend that specific format, this broadcast equipment cannot handle the specifically formatted file.

[0011]

Further, in a case where an attempt is made to record a specifically formatted file on a recording medium, when the formatting system is incompatible with the recording medium, there may be a need, for example, for reading or writing data larger than the file to be read or written in order to read or write such file on the recording medium.

[0012]

The present invention has been made in view of the above circumstances, and therefore it is an object of the present invention to, for example, improve the usability of recording media such as fast reading a header, a body, or a footer constituting a file and to achieve efficient read and write processes by decreasing operations of reading or writing unnecessary data while a file is read from or written to a recording medium.

[MEANS FOR SOLVING THE PROBLEMS]

[0013]

A file generation apparatus according to the present invention is characterized by including: a first data generation means for generating second data to be arranged at the beginning of the file; a second data generation means for generating third data to be arranged at the end of the file; and a third data generation means for generating fourth data which allows the data amount of the first, second, or third data to be an integral multiple of a unit of reading or writing to the recording medium by adding the fourth data to the first data, the second data, or the third data.

[0014]

The first data generation means can generate the second data, i.e., a header of the file.

[0015]

The first data generation means can further include a format conversion means for converting the first data into a KLV (Key, Length, Value) structure, and the first data generation means can generate the second data composed of the file's header, and a key and a length arranged between the header and the first data.

[0016]

The third data generation means can generate the fourth data which allows the data amount of the first data divided into N-1 portions to be an integral multiple of a physical unit area of the recording medium, and the total data amount of the first data to be an integral multiple of the unit of reading and writing on the recording medium by adding the fourth data to each of the N-1 portions of the first data toward the beginning out of the first data divided into N portions, where N is an integer.

[0017]

The third data generation means can generate the fourth data for the first data divided into units corresponding to specified reproduction times with video data and audio data for a plurality of channels multiplexed in accordance with the divided units, the fourth data allows the data amount for each of divided units of the first data to be an integral multiple of the unit of reading and writing on the recording medium.

[0018]

The third data generation means can generate the fourth data which allows the data amount totaling partition data for separating divided portions of the first data from each other, metadata contained in each of divided portions of the first data, and the video data to be an integral multiple of the unit of reading and writing on the recording medium.

[0019]

The third data generation means can generate the fourth data which allows the data amount of each of divided portions of the audio data contained in each of divided portions of the first data to be an integral fraction of the unit of reading and writing on the recording medium, and the total data amount of the audio data to be an integral multiple of the unit of reading and writing on the recording medium.

[0020]

A file generation method according to the present invention is characterized by including: a first data generation step of generating second data to be arranged at the beginning of the file; a second data generation step of generating third data to be arranged at the end of the file; and a third data generation step of generating fourth data which allows the data amount of the first, second, or third data to be an integral multiple of a unit of reading or writing to the recording medium by adding the fourth data to the first data, the second data, or the third data.

[0021]

A program according to the present invention is characterized in that the program causes a computer to perform: a first data generation step of generating second data to be arranged at the beginning of the file; a second data generation step of generating third data to be arranged at the end of the file; and a third data generation step of generating fourth data which allows the data amount of the first, second, or third data to be an integral multiple of a unit of reading or writing to the recording medium by adding the fourth data to the first data, the second data, or the third data.

[0022]

A recording medium according to the present invention is characterized in that the recording medium records first data whose data amount is made to correspond to an integral multiple of a unit of reading or writing to the recording medium by adding first additional data to the first data, so that a boundary of the first data matches a boundary of the unit; the recording medium records second data to be arranged at the beginning of the file whose data amount is made to correspond to an integral multiple of the unit by adding second additional data to the second data, so that a boundary of the second data matches a boundary of the unit; and the recording medium records third data to be arranged at the end of the file whose data amount is made to correspond to an integral multiple of the unit by adding third additional data to the third data, so that a boundary of the third data matches a boundary of the unit.

[0023]

The file generation apparatus, the method, and the program according to the present invention generate the second data to be arranged at the beginning of a file and the third data to be arranged at the end of the file. The fourth data is generated, which arrows the data amount of the first, second, or third data to be an integral multiple of a unit of reading or writing to the recording medium by adding the fourth data to the first, second, or third data.

[0024]

On the recording medium according to the present invention, first data whose data amount is made to correspond to an integral multiple of a unit of reading or writing to the recording medium by adding first additional data to the first data, is recorded so that a boundary of the first data matches a boundary of the unit, and second data to be arranged at the beginning of the file whose data amount is made to correspond to an integral multiple of the unit by adding second additional data to the second data, is recorded so that a boundary of the second data matches a boundary of the unit. Further, third data to be arranged at the end of the file whose data amount is made to correspond to an integral multiple of the unit by adding third additional data to the third data, is recorded so that a boundary of the third data matches a boundary of the unit.

[BEST MODE FOR CARRYING OUT THE INVENTION]

[0025]

Hereinafter, embodiments of the present invention will be described. The exemplary correspondence relationships between constituent components recited in the claims and specific examples in the embodiments of the invention are as follows. This description is to confirm that the specific examples for supporting the inventions recited in the claims are recited in the embodiments of the invention. Therefore, a specific example that is recited in the embodiments of the invention but is not recited in this description as an example corresponding to a constituent component does not mean that the specific example does not correspond to the constituent component. On the contrary, a specific example recited here as an example corresponding to a constituent example does not mean that the specific example does not correspond to constituent components other than the constituent component.

[0026]

Further, the description does not mean that all of the inventions recited in the embodiment of the invention are recited in the claims. In other words, this description does not deny the existence of an invention that corresponds to a specific example that is recited in the embodiments of the invention but is not recited in the claims of this application, i.e., an invention to be filed as divisional application or added by an amendment in the future.

[0027]

The file generation apparatus as described in Claim 1 is characterized by including: a first data generation means for generating second data to be arranged at the beginning of the file (for example, a beginning data generation portion 71 in FIG. 10); a second data generation means for generating third data to be arranged at the end of the file (for example, a footer generation portion 66 in FIG. 10); and a third data generation means for generating fourth data which allows the data amount of the first, second, or third data to be an integral multiple of a unit of reading or writing to the recording medium by adding the fourth data to the first data, the second data, or the third data (for example, filler generation portion 67 in FIG. 10).

[0028]

The recording medium as described in Claim 10 is characterized in that the recording medium records first data (for example, a value of audio data and a KLV-structured filler in FIG. 6) whose data amount is made to correspond to an integral multiple of a unit of reading or writing to the recording medium by adding first additional data (for example, a KLV-structured filler in FIG. 6) to the first data, so that a boundary of the first data matches a boundary of the unit; the recording medium records second data (for example, a length of a header or audio data in FIG. 6) to be arranged at the beginning of the file whose data amount is made to correspond to an integral multiple of the unit by adding second additional data (for example, a filler added to the header in FIG. 6) to the second data, so that a boundary of the second data matches a boundary of the unit; and the recording medium records third data (for example, a footer in FIG. 6) to be arranged at the end of the file whose data amount is made to correspond to an integral multiple of the unit by adding third additional data (for, a footer added to the filler in FIG. 6) to the third data, so that a boundary of the third data matches a boundary of the unit.

[0029]

FIG. 1 shows an exemplary configuration of an embodiment of an AV network system (the system refers to a logical aggregate of multiple apparatuses independently of whether or not constituent apparatuses are included in the same container) to which the present invention is applied.

[0030]

A disk apparatus 1 is composed of a disk drive apparatus 11, a format conversion portion 12, and a communication I/F 13, receives a file of AV data transmitted via a network 4, and records the file on an optical disk 7. Further, the disk apparatus 1 reads the file of AV data recorded on the optical disk 7 and transmits it via the network 4.

[0031]

That is, the optical disk 7 can be mounted to and demounted from the disk drive apparatus 11. The disk drive apparatus 11 drives the optical disk 7 mounted thereon to thus record a file having an AV independent format file described later supplied from the format conversion portion 12 on the optical disk 7. The disk drive apparatus 11 reads the AV independent format file from the optical disk 7 and supplies it to the format conversion portion 12.

[0032]

The format conversion portion 12 is supplied with the AV independent format file from the disk drive apparatus 11, converts this file into a file having a standard AV multiplexing format to be described, and supplies the converted file to the communication I/F 13. The format conversion portion 12 converts the standard AV multiplexing format file supplied from the communication I/F 13 into an AV independent format file, and supplies it to the disk drive apparatus 11.

[0033]

The communication I/F 13 is composed of, for example, an IEEE (Institute of Electrical and Electronics Engineers) 1394 port, a USB (Universal Serial Bus) port, an NIC (Network Interface Card) for LAN (Local Area Network) connection, an analog modem, a TA (Terminal Adapter), a DSU (Digital Service Unit), an ADSL (Asymmetric Digital Subscriber Line) modem, and the like. The communication I/F 13 exchanges standard AV multiplexing format files via the network 4 such as the Internet and intranets. That is, the communication I/F 13 transmits standard AV multiplexing format files supplied from the format conversion portion 12 via the network 4, receives standard AV multiplexing format files transmitted via the network 4, and supplies the files to the format conversion portion 12.

[0034]

In the disk apparatus 1 according to the above-mentioned configuration, the communication I/F 13 receives a standard AV multiplexing format file transmitted via the network 4 and supplies the received file to the format conversion portion 12. The format conversion portion 12 converts the standard AV multiplexing format file from the communication I/F 13 into an AV independent format file and supplies this file to the disk drive apparatus 11. The disk drive apparatus 11 records the AV independent format file from the format conversion portion 12 on the optical disk 7 mounted thereon.

[0035]

Further, the disk apparatus 1 reads the AV independent format file from the optical disk 7 mounted thereon and supplies the file to the format conversion portion 12. The format conversion portion 12 converts the AV independent format file from the disk drive apparatus 11 into a standard AV multiplexing format file and supplies this file to the communication I/F 13. The communication I/F 13 transmits the standard AV multiplexing format file from the format conversion portion 12 via the network 4.

[0036]

The standard AV multiplexing format is compliant with the MXF standard, for example, and is composed of a header, a body, and a footer. Since the standard AV multiplexing format file is compliant with the MXF standard, AV data, i.e., video data and audio data which are multiplexed in units of frames, for example, is placed in the body of the file.

[0037]

In FIG. 1, AV apparatuses 5 and 6 connected to the network 4 are compliant with the MXF standard and are capable of handling MXF compliant files. Accordingly, the AV apparatuses 5 and 6 can transmit standard AV multiplexing format files to the disk apparatus 1 via the network 4. Further, the AV apparatuses 5 and 6 can receive standard AV multiplexing format files transmitted from the disk apparatus 1 via the network 4. That is, the disk apparatus 1 can exchange standard AV multiplexing format files with the AV apparatus 5 or 6 via the network 4. Moreover, the AV apparatuses 5 and 6 can apply various processes such as streaming reproduction to received standard AV multiplexing format files.

[0038]

Here, apparatuses compliant with the present MXF standard such as the AV apparatuses 5 and 6 are hereafter referred to as standard apparatuses, as appropriate.

[0039]

On the other hand, an AV independent format file is composed of a header, a body, and a footer, similarly to a standard AV multiplexing format file. However, only the format of the body differs from the standard AV multiplexing format. That is, in the AV independent format, video data and audio data are separate files. A video file, i.e., a file for video data, has a header and a footer similarly to standard AV multiplexing format files, but in a body of the video file, video data is collectively placed. An audio file, i.e., a file for audio data, has a header and a footer similarly to standard AV multiplexing format files, but in a body of the audio file, audio data is collectively contained.

[0040]

Therefore, in a case where the disk apparatus 1 transmits an AV independent format video file or audio file to the AV apparatus 5 or 6, the AV apparatus 5 or 6 as the standard apparatus, unless compliant with the AV independent format, cannot handle video data or audio data placed in the body of the AV independent format video file or audio file. However, the AV apparatus 5 or 6 can handle the AV independent format video file or audio file itself. In other words, similarly to a standard AV multiplexing format file, the AV independent format video file or audio file is composed of the header, the body, and the footer. The header and the footer have the same format as that of standard AV multiplexing format files. Unless the body "content" (data placed in the body) is referenced, the AV independent format video file or audio file itself is equivalent to a standard AV format file (i.e., standard AV format compliant file). Therefore, even when the AV apparatus 5 or 6 as the standard apparatus is incompatible with the AV independent format, the AV apparatus 5 or 6 can handle the AV independent format video file or audio file itself.

[0041]

That is, only exchange of AV independent format files is available between the disk apparatus 1 and the standard apparatuses such as AV apparatuses 5 and 6.

[0042]

As mentioned above, unless the body "content" is referenced, the AV independent format file is equivalent to the standard AV multiplexing format file. From this viewpoint, AV independent format files are compatible with standard AV multiplexing format files.

[0043]

Next, in FIG. 1, the optical disk 7 can be mounted or dismounted from the disk apparatus 2. Similarly to the AV apparatuses 5 and 6, for example, the disk apparatus 2 is a standard apparatus. From the optical disk 7 mounted thereon, the disk apparatus 2 reads an AV independent format video file or audio file and supplies it to an editing apparatus 3.

[0044]

That is, as mentioned above, unless the body "content" is referenced, the AV independent format video file or audio file is equivalent to a standard AV multiplexing format file. Thus, the disk apparatus 2 as the standard apparatus can read AV independent format video files or audio files from the optical disk 7.

[0045]

The editing apparatus 3 is compatible with the AV independent format and is capable of handling AV independent format files. For example, the editing apparatus 3 performs AV independent editing for AV independent format video files or audio files supplied from the disk apparatus 2 and supplies the disk apparatus 2 with the edited AV independent format video files or audio files.

[0046]

The disk apparatus 2 records the AV independent format video files or audio files supplied from the editing apparatus 3 on the mounted optical disk 7.

[0047]

That is, as mentioned above, unless the body "content" is referenced, the AV independent format video file or audio file is equivalent to a standard AV multiplexing format file. Thus, the disk apparatus 2 as the standard apparatus can record AV independent format video files or audio files on the optical disk 7.

[0048]

As mentioned above, in the standard AV multiplexing format file, video data and audio data are multiplexed in units of frames, for example, and placed in the body of the file. By contrast, in the case of the AV independent format video file or audio file, video data and audio data are collectively placed in the body of the file. This can facilitate editing such as AV independent editing. Further, the AV independent format file uses the header and the footer having the same format as the standard AV multiplexing format file. Thus, unless the body "content" is referenced, AV independent format files are compatible with standard AV multiplexing format files and therefore can be processed on the standard apparatus.

[0049]

Next, FIG. 2 shows an example of the standard AV multiplexing format.

[0050]

FIG. 2 shows the standard AV multiplexing format in a case where video data encoded with MPEG (Moving Picture Experts Group) IMX called D10 and audio data uncompressed according to the AES (Audio Engineering Society) 3 format are employed as video data and audio data placed in the body.

[0051]

In addition, other video data and audio data according to various formats such as DV (Digital Video) may be placed in the body.

[0052]

A standard AV multiplexing format file is composed of a header (File Header), a body (File Body), and a footer (File Footer) in the stated order.

[0053]

In the header, Header Partition Pack, Header Metadata, and Index Table are placed in order from the beginning. In the header partition pack, data for specifying the header, information indicative of a format and a file format of the data placed in the body, and the like are placed. In the header metadata, file-based metadata such as a file creation date and information about the data placed in the body is placed, for example. In the index table, a table indicative of a location of an edit unit (described later) placed in the body is placed.

[0054]

The metadata includes a time code supplied to the video file for each frame or the like, UMID (Unique Material Identifier), GPS (Global Positioning System) information representing positions of capturing by a video camera, capturing date and time (year, month, day, hour, minute, second), ARIB (Association of Radio Industries and Businesses) metadata, and setting or control information about the video camera for capturing. ARIB metadata is standardized by ARIB and is superposed on a standard communication interface such as SDI (Serial Digital Interface). The setting or control information about the video camera includes, for example, IRIS control values, white-balance or black-balance mode, and lens information about lens zooms and focuses.

[0055]

The index table is optional and may or may not be included in the header. Further, in addition to the index table various optional data may be placed in the header.

[0056]

In addition, in a case of the standard AV multiplexing format file, information indicative of the standard AV multiplexing format is employed as the information indicative of the file format placed in the header partition pack. Meanwhile, in a case of the AV independent format file, information indicative of the AV independent format is employed. It should be noted that the header partition pack format itself is unchanged for the standard AV multiplexing format and the AV independent format.

[0057]

The footer is composed of Footer Partition Pack in which data for specifying the footer and the like is placed.

[0058]

The body is composed of one or more edit units. The edit unit signifies a unit per frame, and AV data and the like for one frame is placed in the edit unit.

[0059]

That is, the edit unit is composed of a system item, a picture item, a sound item, and an auxiliary item which are placed in order from the beginning.

[0060]

In the system item, metadata (frame-based metadata) concerning a frame of video data placed in the succeeding picture item is placed. Here, the frame-based metadata includes time codes, for example.

[0061]

In the picture item, video data for one frame is placed. In FIG. 2, the above-mentioned D10 formatted video data is placed in the picture item.

[0062]

Here, in the picture item, video data for one frame subjected to KLV coding in accordance with the KLV (Key, Length, Value) structure is placed.

[0063]

The KLV structure refers to a structure in which a key, a length, and a value are sequentially placed from the beginning in order. In the key, an SMPTE 298M standard compliant 16-byte label indicating what kind of data is contained in the value is placed. In the length, the length of the data placed in the value is placed. In the value, actual data, i.e., video data for one frame in this example is placed.

[0064]

In addition, the data length of the picture item is a fixed data length based on KAG (KLV Alignment Grid). For the picture item to have the fixed length, a filler as stuffing data that is also in accordance with the KLV structure is placed after the video data of the picture item.

[0065]

The fixed length based on KAG, i.e., the picture item's data length is expressed in integral multiples (e.g., 512 bytes, 2 kilobytes, and the like) of the sector length of the optical disk 7, for example. In this case, the affinity between the optical disk 7 and the picture item increases, making it possible to accelerate operations to read and write the picture item on the optical disk 7.

[0066]

Similarly to the picture item, in the system item described above as well as the sound item and the auxiliary item described later, the KLV structure is employed, and the data lengths of those items are the fixed data lengths based on the KAG.

[0067]

In the sound item, audio data corresponding to one frame of the video data placed in the picture item is placed according to the KLV structure, similarly to the above-mentioned picture item.

[0068]

In the sound item, multiple pieces of audio data, e.g., eight channels of audio data are placed in a multiplexed fashion.

[0069]

That is, in the sound item, the value of the KLV structure sequentially contains Element Header EH, Audio Sample Count ASC, Stream Valid Flags SVF, and multiplexed 8-channel audio data in order from the beginning.

[0070]

In the sound item, the 8-channel audio data is multiplexed by arranging audio data samples such as a first sample, a second sample, and so on in order for each of eight channels in one frame. In the audio data shown at the bottom of FIG. 2, a number in parentheses indicates to which ordinal position the audio data sample corresponds.

[0071]

In the element header EH, data to specify the element header and the like is placed. In the audio sample count ASC the number of audio data samples placed in the sound item is placed. The stream valid flag SVF is an eight-bit (one-byte) flag, each bit of which represents whether or not audio data for the channel corresponding to that bit is valid. That is, each bit of the stream valid flag SVF is set to 1 for example when audio data for the channel corresponding to the bit is valid, and set to 0 for example when audio data for the channel corresponding to the bit is invalid.

[0072]

In the auxiliary item, necessary user data is placed. Accordingly, the auxiliary item is an area where a user can place any type of data.

[0073]

As mentioned above, in the standard AV multiplexing format, the system item in which frame-based metadata is placed, the picture item in which video data is placed, the sound item in which audio data is placed, and the auxiliary item in which user data is placed are multiplexed in units of frames. Further, in the sound item, 8-channel audio data is multiplexed in units of samples.

[0074]

Thus, in a case of a file in which video data and audio data are separately and collectively placed, the video data and the audio data cannot be reproduced until all the files for the video data and the audio data are received. By contrast, in the standard AV multiplexing format, since video data and audio data are multiplexed in units of frames, receiving video data and audio data for one frame makes it possible to promptly reproduce the video data and the audio data for the frame. Accordingly, the standard AV multiplexing format is suited for streaming.

[0075]

As mentioned above, the standard AV format in which video data and audio data are multiplexed in units of frames is suited for streaming. However, on the negative side, AV independent editing to separately edit video data and audio data is difficult in the standard AV multiplexing format.

[0076]

Further, the file-based metadata is also scattered in the system item of the edit unit, and is difficultly handled during editing and the like.

[0077]

Furthermore, the AES3 standard which can be employed in the standard AV format has the specification of allocating at least four bytes to one sample of audio data, this increases the size of the entire file.

[0078]

In this regard, FIG. 3 shows an example of the AV independent format.

[0079]

In the AV independent format, video data, audio data, file-based metadata, and user data are independently and collectively arranged, while these data are multiplexed in the standard AV multiplexing format.

[0080]

That is, in the AV independent format, picture items that contain video data according to the standard AV multiplexing format. Further, the body is provided with a header and a footer having the same format as the standard AV multiplexing format to constitute a video file.

[0081]

It should be noted that, in the body of the video file according to the AV independent format, picture items as many as an integral multiple of the sector length for the optical disk 7 are collectively placed. Accordingly, the size of the entire body is equivalent to an integral multiple of the sector length for the optical disk 7. That is, the body of the video file according to the AV independent format is sized in accordance with sector alignment.

[0082]

Further, the entire body of the video file is sized to an integral multiple of the ECC block length for the optical disk 7. As will be described later, the last filler in the body is adjusted in size so that the entire body of the video file is sized to an integral multiple of the ECC (Error Correction Code) block length for the optical disk 7.

[0083]

The ECC block is a unit of performing the ECC process as a unit of reading and writing on the optical disk 7.

[0084]

The sector is an example of physical unit areas on the optical disk 7. The ECC block is an example of units for reading and writing on the optical disk 7. In addition, the fixed number of sectors can be used as a physical unit area on the optical disk 7. A unit of reading and writing on the optical disk 7 can be the fixed number of physical unit areas, for example.

[0085]

Herein, the ECC process is performed in a signal process portion 115 (described later) in units of ECC blocks, for example. An ECC block can be composed of one or more sectors.

[0086]

In the following description, one sector is assumed to be a physical unit area on the optical disk 7, and one ECC block is assumed to be a unit of reading and writing composed of one or more sectors.

[0087]

In FIG. 2, although the index table is shown in the header of the standard AV multiplexing format file, the index table is optional in the MXF, as described above. Thus, no index table is employed in a video file in FIG. 3 (also in an audio file described later).

[0088]

In the AV independent format, multiplexed 8-chanel audio data, which is placed in the sound item in the standard AV multiplexing format, is separated into channel-based audio data, converted into the WAVE format from the AES3 format, and then stored in the file's body for each channel according to the KLV structure. Further, the body of the file is added with a header and a footer having the same format as the standard AV multiplexing format, to thereby constitute an audio file.

[0089]

That is, in the AV independent format, an audio file corresponding to each channel of the 8-channel audio data is independently formed. The audio file for each channel is constituted by placing WAVE-formatted and collectively KLV-structured audio data for the channel in the body of the file, and adding a header and a footer having the same format as the standard AV multiplexing format to the body.

[0090]

As mentioned above, the body of the audio file according to the AV independent format contains WAVE-formatted and collectively KLV-structured audio data for a given channel. However, the entire audio data is not always sized to an integral multiple of the ECC block length for the optical disk 7. In this regard, the body of the audio file according to the AV independent format is provided with not only a KLV-structured filler after the KLV-structured audio data, but also fillers after the header and the footer.

[0091]

In the AV independent format, in addition to the video file and the audio file for each of eight channels as mentioned above, a file-based metadata file that collectively contains file-based metadata contained in the header metadata according to the standard AV multiplexing format, and a frame-based metadata file that collectively contains system items containing the frame-based metadata according to the standard AV multiplexing format are provided. Further, in the AV independent format, an auxiliary file collectively containing user data according to the standard AV multiplexing format is provided.

[0092]

In the AV independent format, a master file is provided, which describes pointers to the video file, the audio files corresponding to eight channels, the file-based metadata file, the frame-based metadata file, and the auxiliary file.

[0093]

That is, the master file is coded in XML (Extensible Markup Language), for example, and contains, e.g., file names of respective files as pointers to the video file, the audio files corresponding to eight channels, the file-based metadata file, the frame-based metadata file, and the auxiliary file.

[0094]

Accordingly, the master file can be used to reference the video file, the audio files corresponding to eight channels, the file-based metadata file, the frame-based metadata file, and the auxiliary file.

[0095]

It should be noted that the auxiliary file may be optional, for example.

[0096]

Further, in FIG. 3, the file-based metadata file, the frame-based metadata file, and the auxiliary file do not have the header and the footer having the same format as the standard AV multiplexing format. However, the file-based metadata file, the frame-based metadata file, and the auxiliary file may also be configured by adding thereto the header and the footer having the same format as the standard AV multiplexing format.

[0097]

Furthermore, a minimum set of file-based metadata is placed in the header metadata that constitutes headers for the video file and the audio files according to the AV independent format.

[0098]

That is, in the AV independent format, there exists the file-based metadata file in which file-based metadata, which is placed in the header metadata in the standard AV multiplexing format, is collectively placed. Therefore, it is redundant to store the file-based metadata contained in the metadata file in addition to the header metadata that provides headers for the video file and the audio files. This also increases the size of the AV independent format file.

[0099]

However, in the MXF, the header metadata is a necessary item for headers. When a header is configured without using any header metadata, the header will not have that same format as that in the standard AV multiplexing format.

[0100]

On the other hand, in the MXF, various items are available for file-based metadata to be stored in the header metadata, some of these items are necessary and the others are optional.

[0101]

In this regard, in order to prevent the file size from increasing and maintain the compatibility with the standard AV multiplexing format, only a minimum set of file-based metadata, i.e., metadata for items which are necessary to be placed in the header metadata in the MXF, is placed in the header metadata that constitutes headers for the video file and the audio files according to the AV independent format.

[0102]

As mentioned above, in the AV independent format, video data is collectively placed in the video file and audio data for each channel is collectively placed in the audio file corresponding to the channel. Accordingly, it is possible to easily perform editing such as AV independent editing to separately edit video data and audio data.

[0103]

Further, since the AV independent format stores audio data in the WAVE format, the data amount can be reduced compared to AES3 audio data according to the standard AV independent format. As a result, when an AV independent format file is recorded in a storage such as the optical disk 7, the storage capacity needed for the recording can be saved compared to recording of a standard AV multiplexing format file.

[0104]

Further, similarly to standard AV multiplexing format files, AV independent format video files and audio files are each configured to be provided with the header, the body, and the footer in order from the beginning. Moreover, the header and the footer have the same format as the standard AV multiplexing format. Accordingly, in a case where the disk apparatus 1 records an AV independent format video file or audio file on the removable optical disk 7 and then the optical disk 7 is mounted on the disk apparatus 2, when the disk apparatus 2 is a standard apparatus (capable of handling MXF files), the AV independent format video file or audio file can be read from the optical disk 7.

[0105]

Furthermore, in the AV independent format, file-based metadata and frame-based metadata are each independently collected to be one file, thereby facilitating a retrieval process using metadata.

[0106]

FIGS. 4 and 5 are diagrams illustrating the data amount of an AV independent format video file. As shown in FIG. 4, a filler is placed after the header of the AV independent format video file, and the entire header is sized to an integral multiple of the ECC block length for the optical disk 7. The video file is written to the optical disk 7 so that the video file's header boundary corresponds to the ECC block boundary on the optical disk 7.

[0107]

A filler is placed after the video file's footer. The entire footer is sized to an integral multiple of the ECC block length for the optical disk 7. The video file is written to the optical disk 7 so that the video file's footer boundary corresponds to the ECC block boundary on the optical disk 7.

[0108]

The entire body of the video file is sized to an integral multiple of the ECC block length for the optical disk 7. The video file is written to the optical disk 7 so that the body's boundary corresponds to the ECC block boundary on the optical disk 7. Further, each picture item of the body and the succeeding filler are an integral multiple of the sector length for the optical disk 7. The video file is written to the optical disk 7 so·that the preceding boundary of the picture item corresponds to the sector boundary and the boundary succeeding to the filler added to the picture item corresponds to the sector boundary.

[0109]

As shown in FIG. 5, the last filler in the body is adjusted in size so that the entire body is sized to an integral multiple of the ECC block length for the optical disk 7. When a video file is written to the optical disk 7, the boundary succeeding to the filler added to the last picture item in the body corresponds to the ECC block boundary.

[0110]

FIG. 6 is a diagram illustrating the data amount of an AV independent format audio file. The filler at the end of the header is adjusted in size so that the audio file's header and the audio data's key and length according to the KLV structure in the body are sized to an integral multiple of the ECC block length for the optical disk 7. The audio file is written to the optical disk 7 so that the boundary preceding the audio file's header corresponds to the boundary of the ECC block for the optical disk 7. Further, the audio file is written to the optical disk 7 so that the boundary succeeding the length corresponds to the boundary of the ECC block for the optical disk 7.

[0111]

The KLV-structured value of the audio data in the body and the KLV-structured filler added to the body are sized to an integral multiple of the ECC block length for the optical disk 7. The audio file is written to the optical disk 7 so that the boundary succeeding the body corresponds to the boundary of the ECC block for the optical disk 7.

[0112]

The audio file's footer is followed by a filler. The entire footer is sized to an integral multiple of the ECC block length for the optical disk 7. The audio file is written to the optical disk 7 so that the boundaries preceding and succeeding the audio file's footer correspond to the boundaries of the ECC block for the optical disk 7.

[0113]

FIG. 7 shows an exemplary configuration of the format conversion portion 12 included in the disk apparatus 1 in FIG. 1.

[0114]

The format conversion portion 12 is composed of a standard/independent format conversion portion 21 and an independent/standard format conversion portion 22.

[0115]

The standard/independent format conversion portion 21 converts a file according to the standard AV multiplexing format in FIG. 2 supplied from the communication I/F 13 into a file according to the AV independent format in FIG. 3, and supplies it to the disk drive apparatus 11. The independent/standard format conversion portion 22 converts the file according to the AV independent format in FIG. 3 supplied from the disk drive apparatus 11 into a file according to the standard AV multiplexing format in FIG. 2, and supplies it to the communication I/F 13.

[0116]

FIG. 8 shows an exemplary configuration of the standard/independent format conversion portion 21 in FIG. 7.

[0117]

A buffer 31 is supplied with the standard AV multiplexing format file from the communication I/F 13. The buffer 31 temporarily stores the supplied standard AV multiplexing format file.

[0118]

When the standard AV multiplexing format file is stored in the buffer 31, the master file generation portion 32 generates a master file according to the AV independent format for the standard AV multiplexing format file and supplies it to a buffer 44.

[0119]

The header acquisition portion 33 acquires a header by extracting it from the standard AV multiplexing format file stored in the buffer 31 and supplies the header to a header metadata extraction portion 35.

[0120]

The body acquisition portion 34 acquires a body by extracting it from the standard AV multiplexing format file stored in the buffer 31, and supplies the body to a system item process portion 36, an auxiliary item extraction portion 38, a picture item extraction portion 40, and a sound item extraction portion 42.

[0121]

The header metadata extraction portion 35 extracts header metadata from the header supplied from the header acquisition portion 33, and supplies a metadata file generation portion 37 with file-based metadata placed in the header metadata. The system item process portion 36 extracts the system item in which frame-based metadata is placed from each edit unit in the body supplied from the body acquisition portion 34, and supplies the system item to the metadata file generation portion 37. The metadata file generation portion 37 generates a file-based metadata file in which the file-based metadata supplied from the header metadata extraction portion 35 is placed. The metadata file generation portion 37 also generates a frame-based metadata file in which system items in respective edit units supplied from the system item process portion 36 are collectively (sequentially) placed, and supplies the file-based metadata file and the frame-based metadata file to the buffer 44.

[0122]

The auxiliary item extraction portion 38 extracts an auxiliary item in which frame-based user data is placed from each edit unit in the body supplied from the body acquisition portion 34 and supplies the auxiliary item to the auxiliary file generation portion 39. The auxiliary file generation portion 39 generates an auxiliary file in which auxiliary items in the edit units supplied from the auxiliary item extraction portion 38 are collectively placed, and supplies the auxiliary file to the buffer 44.

[0123]

The picture item extraction portion 40 extracts a picture item in which frame-based video data is placed from each edit unit in the body supplied from the body acquisition portion 34 and supplies the picture item to a video file generation portion 41. The video file generation portion 41 collectively stores picture items in the edit units supplied from the picture item extraction portion 40. Further, the video file generation portion 41 generates a video file whose body is added with the header and the footer having the same format as the standard AV multiplexing format and supplies the video file to the buffer 44.

[0124]

The sound item extraction portion 42 extracts a sound item in which frame-based audio data is placed from each edit unit in the body supplied from the body acquisition portion 34 and supplies the sound item to an audio file generation portion 43. The audio file generation portion 43 generates an audio file for each channel obtained by collectively placing, in the audio file's body, channel-based audio data placed in the sound item of each edit unit supplied from the sound item extraction portion 42, and adding the header and the footer having the same format as the standard AV multiplexing format to the body, and supplies the audio file to the buffer 44.

[0125]

The buffer 44 temporarily stores the master file supplied from the master file generation portion 32, the file-based metadata file and the frame-based metadata file supplied from the metadata file generation portion 37, the auxiliary file supplied from the auxiliary file generation portion 39, the video file supplied from the video file generation portion 41, and the channel-based audio file supplied from the audio file generation portion 43, and supplies these files as AV independent format files to the disk drive apparatus 11.

[0126]

FIG. 9 shows an exemplary configuration of the video file generation portion 41 in FIG. 8.

[0127]

Picture items in the edit units supplied from the picture item extraction portion 40 are supplied to a connection portion 51. The connection portion 51 sequentially connects (concatenates) the supplied picture items in the edit units and supplies the connected picture items to a footer generation portion 52. The footer generation portion 52 generates a footer having the same format as the standard AV multiplexing format file, the footer is to be added to a body obtained by connecting the picture items supplied from the connection portion 51. The footer generation portion 52 supplies the footer and the body to a header generation portion 53.

[0128]

The header generation portion 53 generates a header to be added to the footer and the body supplied from the footer generation portion 52. The header generation portion 53 supplies the header, the body, and the footer to a filler generation portion 54.

[0129]

The filler generation portion 54 generates a filler to be added to the header and a filler to be added to the footer. Further, the filler generation portion 54 generates a last filler in the body. A KLV encoder 55 in the filler generation portion 54 encodes the body's last filler according to the KLV structure.

[0130]

The filler generation portion 54 configures an AV independent format video file composed of the header, the body, and the footer added with fillers and outputs the video file.

[0131]

By adding a filler generated by the filler generation portion 54 to the header, the body, or the footer in the video file, the data amounts of the header, the body, and the footer are adjusted to be integral multiples of the ECC block length for the optical disk 7.

[0132]

With this structure, when the video file is written to the optical disk 7, it becomes possible to prevent the header, the body, or the footer from being recorded in a part of the ECC block, and more efficiently read and write the video file.

[0133]

Since each of the header, the body, and the footer is an integral multiple of the ECC block length for the optical disk 7, by recording the header, the body, and the footer so that their boundaries correspond to the ECC block boundaries, the header, the body, or the footer can be write or read to/from the minimum number of ECC blocks when only the header, the body, or the footer is to be written or read. That is, it becomes possible to more efficiently read or write video files from/to the optical disk 7.

[0134]

FIG. 10 shows an exemplary configuration of the audio file generation portion 43 in FIG. 8.

[0135]

The sound item in each edit unit supplied from the sound item extraction portion 42 is supplied to a KLV decoder 61. The KLV decoder 61 decomposes the KLV structure of the audio data placed in the sound item of each edit unit to generate audio data for eight multiplexed channels (hereafter referred to as multiplexed audio data as appropriate). The KLV decoder 61 supplies the resulting multiplexed audio data to a channel separation portion 62.

[0136]

The channel separation portion 62 separates channel-based audio data from the multiplexed audio data for each sound item supplied from the KLV decoder 61. The channel separation portion 62 groups the channel-based audio data for each channel and supplies it to the data conversion portion 63.

[0137]

The data conversion portion 63 converts the coding system for channel-based audio data supplied from the channel separation portion 62. That is, the standard AV multiplexing format uses AES3-coded audio data. The AV independent format uses WAVE-coded audio data. For this reason, the data conversion portion 63 converts AES3-coded audio data (AES3-system audio data) supplied from the channel separation portion 62 into WAVE-coded audio data (WAVE-system audio data).

[0138]

In the above-mentioned example, the data conversion portion 63 converts AES3-system audio data into WAVE-system audio data. The data conversion portion 63 may convert audio data into any audio data other than the WAVE system. That is, the data conversion portion 63 converts audio data for the purpose of suppressing the amount of AES3-system audio data. Thus, the data conversion portion 63 may use any coding system that can achieve the above-mentioned purpose.

[0139]

The audio file generation portion 43 may be configured without the data conversion portion 63 when the amount of audio data is omissible.

[0140]

Channel-based WAVE-system audio data obtained in the data conversion portion 63 is supplies to a KLV encoder 64. The KLV encoder 64 KLV-codes each of the audio data grouped into channels according to the KLV structure, and supplies it to a header generation portion 65.

[0141]

The header generation portion 65 generates a header having the same format as that of standard AV multiplexing format files, the header is to be added to each channel's body composed of each channel-based audio data supplied from the KLV encoder 64, and supplies the body and the header to a footer generation portion 66.

[0142]

The footer generation portion 66 generates a footer having the same format as that of standard AV multiplexing format files, the header is to be added to the body. The footer generation portion 66 supplies the header, the footer, and the body to the filler generation portion 67.

[0143]

The filler generation portion 67 generates a filler to be added to the header, a filler to be added to the body, and a filler to be added to the footer. Here, as shown in FIG. 6, the filler generation portion 67 generates a filler so that data amounts of the header, and the key and the length added by the KLV encoder 64 match integral multiples of the data amount of the ECC block, and adds the generated filler after the header. Further, as shown in FIG. 6, the filler generation portion 67 generates a filler so that the data amount of the footer matches an integral multiple of the data amount of the ECC block and adds the generated filler after the footer.

[0144]

A KLV encoder 68 in the filler generation portion 67 encodes a filler to be added to the body in accordance with the KLV structure. As shown in FIG. 6, the filler generation portion 67 generates a filler encoded in the KLV structure so that the data amount of audio data matches an integral multiple of the data amount of the ECC block, and adds the generated filler after the audio data.

[0145]

The KLV encoder 64 and the header generation portion 65 constitutes a beginning data generation portion 71.

[0146]

In this manner, by adding fillers generated by the filler generation portion 54 to the header, the audio data, or the footer, data amounts of the header, and the key and the length added by the KLV encoder 64, the audio data, and the footer are adjusted to be integral multiples of the ECC block length for the optical disk 7.

[0147]

In this manner, when an audio file is written to the optical disk 7, it is possible to prevent the header, the body, or the footer from being recorded in a part of the ECC block, and more efficiently read and write video files.

[0148]

Since each of the header, and the key and the length added by the KLV encoder 64, the audio data, and the footer is an integral multiple of the ECC block length for the optical disk 7, by recording the header, and the key and the length added by the KLV encoder 64, the audio data, or the footer so that each boundary of these matches the ECC block boundary, the header, and the key and the length added by the KLV encoder 64, the audio data, or the footer can be read or written from or to the minimum number of ECC blocks when only the header, and the key and the length added by the KLV encoder 64, the audio data, or the footer is to be read. That is, it becomes possible to more efficiently read and write audio files to the optical disk 7.

[0149]

The standard/independent format conversion portion 21 in FIG. 8 performs a master file generation process to generate a master file as an AV independent format file, a metadata file generation process to generate a file-based metadata file and a frame-based metadata file, an auxiliary file generation process to generate an auxiliary file, a video file generation process to generate a video file, and an audio file generation process to generate an audio file.

[0150]

With reference to flowcharts in FIGS. 11 through 13, the following describes the master file generation process, the metadata file generation process, the auxiliary file generation process, the video file generation process, and the audio file generation process performed by the standard/independent format conversion portion 21.

[0151]

First, the master file generation process will be described with reference to the flowchart in FIG. 11.

[0152]

The master file generation process starts when a standard AV format file is supplied and is stored in the buffer 31 (FIG. 8), for example. First, at Step S1, the master file generation portion 32 (FIG. 8) generates file names of a file-based metadata file, a frame-based metadata file, an auxiliary file, a video file, and audio files for respective channels, and the process proceeds to Step S2. At Step S2, the master file generation portion 32 generates a master file in which links to the file names generated at Step S1 are described in the XML code, and supplies the links to the buffer 44 for storing the links. The master file generation process then terminates.

[0153]

Next, referring to the flowchart in FIG. 12, the file-based metadata file generation process to generate file-based metadata files will be described.

[0154]

The file-based metadata file generation process starts when a standard

AV format file is supplied and stored in the buffer 31 (FIG. 8), for example. At Step S11, the header acquisition portion 33 first obtains a header from the standard AV format file stored in the buffer 31 and supplies the header to the header metadata extraction portion 35, and the process proceeds to Step S12. At Step S12, the header metadata extraction portion 35 extracts metadata from the header supplied from the header acquisition portion 33, and supplies file-based metadata placed in the metadata to the metadata file generation portion 37. The process proceeds to Step S13. At Step S13, the metadata file generation portion 37 generates a file-based metadata file in which the file-based metadata supplied from the header metadata extraction portion 35 is placed, and supplies the file to the buffer 44 for storing it. The file-based metadata file generation process then terminates.

[0155]

Next, referring to the flowchart in FIG. 13, the following describes the frame-based metadata file generation process to generate frame-based metadata files.

[0156]

The frame-based metadata file generation process starts when a standard AV format file is supplied and stored in the buffer 31 (FIG. 8), for example. At Step S21, the body acquisition portion 34 first obtains a body from the standard AV multiplexing format file stored in the buffer 31 and supplies the body to the system item process portion 36. The process proceeds to Step S22. At Step S22, the system item process portion 36 extracts the system item in which frame-based metadata is placed from each edit unit in the body supplied from the body acquisition portion 34 and supplies the system item to the metadata file generation portion 37. The process proceeds to Step S23. At Step S23, the metadata file generation portion 37 is supplied with the system item for each edit unit from the system item process portion 36 and adds a filler to the system item. The process proceeds to Step S24.

[0157]

At Step S24, the metadata file generation portion 37 connects the system items to which the fillers are added to generate a body of a file-based metadata in which the system items for the respective edit units are collectively placed, and supplies the generated body to the buffer 44. The process proceeds to Step S25. At Step S25, the buffer 44 outputs the metadata file body. The process proceeds to Step S26.

[0158]

At Step S26, the metadata file generation portion 37 generates a footer, and the process proceeds to Step S27. At Step S27, the metadata file generation portion 37 generates a footer's filler and supplies the footer to which the filler is added to the buffer 44. The process proceeds to Step S28. At Step S28, the buffer 44 outputs the footer, and the process proceeds to Step S29.

[0159]

At Step S29, the metadata file generation portion 37 generates a header, and the process proceeds to Step S30. At Step S27, the metadata file generation portion 37 generates a header's filler and supplies the filler-provided header to the buffer 44. The process proceeds to Step S31. At Step S31, the buffer 44 outputs the header, and the frame-based metadata file generation process then terminates.

[0160]

Next, referring to the flowchart in FIG. 14, the auxiliary file generation process to generate auxiliary files will be described.

[0161]

The auxiliary file generation process starts when a standard AV format file is supplied and stored in the buffer 31 (FIG. 8), for example. At Step S41, the body acquisition portion 34 first obtains a body from the standard AV multiplexing format file stored in the buffer 31 and supplies the body to the auxiliary item extraction portion 38. The process proceeds to Step S42. At Step S42, the auxiliary item extraction portion 38 extracts an auxiliary item from each edit unit of the body supplied from the body acquisition portion 34 and supplies the auxiliary item to the auxiliary file generation portion 39. The process proceeds to Step S43. At Step S43, the auxiliary file generation portion

39 connects auxiliary items for respective edit units supplied from the auxiliary item extraction portion 38 to generate an auxiliary file in which the auxiliary items for the edit units are collectively placed. The auxiliary file control portion 39 supplies the auxiliary file to the buffer 44 so as to store it. The auxiliary file generation process then terminates.

[0162]

Next, referring to the flowchart in FIG. 15, the video file generation process to generate video files will be described.

[0163]

The video file generation process starts when a standard AV format file is supplied and stored in the buffer 31 (FIG. 8), for example. At Step S51, the body acquisition portion 34 first obtains a body from the standard AV multiplexing format file stored in the buffer 31 and supplies the body to the picture item extraction portion 40. The process proceeds to Step S52. At Step S52, the picture item extraction portion 40 extracts a picture item from each edit unit of the body supplied from the body acquisition portion 34 and supplies the picture item to the video file generation portion 41. The process proceeds to Step S53. At Step S53, the connection portion 51 in the video file generation portion 41 (FIG. 9) connects picture items for the edit units supplied from the picture item extraction portion 40 to generate a body in which the picture items for the edit units are collectively placed. The process proceeds to Step S54.

[0164]

At Step S54, the video file generation portion 41 determines whether or not the connected picture item is the last one. When the connected picture item is determined to be not the last one, the process proceeds to Step S55. The video file generation portion 41 outputs the generated body to the buffer 44, and the process returns to step S52 at which the above-mentioned process is repeated. In this case, the body unchangedly passes through the footer generation portion 52, the header generation portion 53, and the filler generation portion 54.

[0165]

When the connected picture item is determined to be the last one at Step

S54, the process proceeds to Step S56. The filler generation portion 54 in the video file generation portion 41 (FIG. 9) generates a filler for the last picture item, whose data amount is adjusted so that the body's data amount matches an integral multiple of the ECC block after conversion into the KLV structure. The process proceeds to Step S57. At Step S57, the KLV encoder 55 converts the last picture item's filler into the KLV structure. The process proceeds to Step S58. At Step S58, the video file generation portion 41 makes a body out of the filler converted into the KLV structure and outputs the body. The process proceeds to Step S59.

[0166]

At Step S59, the footer generation portion 52 generates a footer, and the process proceeds to Step S60. At Step S60, the filler generation portion 54 generates a footer's filler whose data amount is adjusted so that the footer's data amount matches an integral multiple of the ECC block. The process proceeds to Step S61. At Step S61, the video file generation portion 41 outputs the footer, and the process proceeds to Step S62.

[0167]

At Step S62, the header generation portion 53 generates a header, and the process proceeds to Step S63. At Step S63, the filler generation portion 54 generates a header's filler whose data amount is adjusted so that the header's data amount matches an integral multiple of the ECC block. The process proceeds to Step S64. At Step S64, the video file generation portion 41 outputs the header, and the video file generation process then terminates.

[0168]

In this manner, the header is generated after the body and the footer. Consequently, a single process can be used to generate a header containing data such as the video data's reproduction time or time code (TC) that need to be settled by the body.

[0169]

When the header is created first, for example, data such as the video data's reproduction time or time code (TC) is not settled and cannot be written until the body creation terminates. When the body creation terminates, the reproduction time or the time code needs to be written to the header, causing duplication of work. When a video file is recorded on a recording medium such as the optical disk 7, an unnecessary process occurs such as seeking to the header. Alternatively, an unsettled data amount of the header makes it difficult to ensure an area needed to record the header. Depending on conditions, the header may be recorded apart from the body and the footer on the optical disk 7.

[0170]

Generating the header after the body and the footer can eliminate such duplication of work, and a single process can suffice for generating the header containing data that is dependent on settlement of the body. When a video file is recorded on a recording medium such as the optical disk 7, the header can be reliably recorded successively to the body and the footer.

[0171]

Next, referring to the flowchart in FIG. 16, the audio file generation process to generate audio files will be described.

[0172]

The audio file generation process starts when a standard AV format file is supplied and stored in the buffer 31 (FIG. 8), for example. At Step S71, the body acquisition portion 34 first obtains a body from the standard AV multiplexing format file stored in the buffer 31 and supplies the body to the sound item extraction portion 42. The process proceeds to Step S72. At Step S72, the sound item extraction portion 42 extracts a sound item from each edit unit of the body supplied from the body acquisition portion 34 and supplies the sound item to the audio file generation portion 43. The process proceeds to Step S73. At Step S73, the KLV decoder 61 of the audio file generation portion 43 (FIG. 10) decomposes the KLV structure of audio data placed in the sound item of each edit unit, and supplies the channel separation portion 62 with resulting audio data for eight multiplexed channels (multiplexed audio data). The process proceeds to Step S74.

[0173]

At Step S74, the channel separation portion 62 is supplied with the multiplexed audio data for each sound item from the KLV decoder 61 and separates channel-based AES3 audio data from the multiplexed audio data. The channel separation portion 62 groups the channel-based AES3 audio data for each channel and supplies it to the data conversion portion 63.

[0174]

The process proceeds to Step S75. The data conversion portion 63 converts the channel-based AES3 audio data supplied from the channel separation portion 62 into WAVE-system audio data and supplies it to the KLV encoder 64. The process proceeds to Step S76. At Step S76, the KLV encoder 64 KLV-codes WAVE-system audio data grouped for each channel supplied from the data conversion portion 63 according to the KLV structure. In this manner, the KLV encoder 64 generates a channel-based body in which the WAVE-system audio data for each channel is collectively placed. The process proceeds to Step S77.

[0175]

At Step S77, the filler generation portion 67 generates a filler for each channel's body, whose data amount is adjusted so that the data amount of the KLV-structured audio data matches an integral multiple of the ECC block. The process proceeds to Step S78. At Step S78, the KLV encoder 68 KLV-codes each filler for each channel's body according to the KLV structure. The process proceeds to Step S79. At Step S79, the audio file generation portion 43 outputs each channel's body. The process proceeds to Step S80. When each channel's body is output, the audio data's value and the KLV-structured filler are output. The audio data's key and length are not output.

[0176]

At Step S80, the footer generation portion 66 generates a footer for each channel, and the process proceeds to Step S81. At Step S81, the filler generation portion 67 generates a filler for each channel's footer, whose data amount is adjusted so that the footer's data amount matches an integral multiple of the ECC block. The process proceeds to Step S82. At Step S82, the video file generation portion 41 outputs each channel's footer, and the process proceeds to Step S83.

[0177]

At Step S83, the header generation portion 65 generates a header for each channel, and the process proceeds to Step S84. At Step S84, the filler generation portion 54 generates a filler for each channel's header, whose data amount is adjusted so that the data amounts of the header and the audio data's key and length match integral multiples of the ECC block. The process proceeds to Step S85. At Step S85, the video file generation portion 41 outputs each channel's header, and the audio file generation process then terminates. At Step S85, the audio data's key and length are output as well as each channel's header.

[0178]

In this manner, the header is generated after the body and the footer. Consequently, a single process can be used to generate a header containing data such as the audio data's reproduction time or time code (TC) that need to be settled by the body.

[0179]

When an audio file is recorded on a recording medium such as the optical disk 7, the header can be reliably recorded successively to the body and the footer.

[0180]

FIG. 17 shows an exemplary configuration of the disk drive apparatus 11 according to an embodiment of the present invention.

[0181]

A spindle motor 111 operates based on a spindle motor drive signal from a servo control portion 114 to rotatively drive the optical disk 7 at CLV (Constant Linear Velocity) or CAV (Constant Angular Velocity).

[0182]

A pickup portion 112 controls laser beam output based on a recording signal supplied from the signal process portion 115 to record the recording signal on the optical disk 7. The pickup portion 112 also condenses and radiates a laser beam on the optical disk 7, and photoelectrically converts the reflected light from the optical disk 7 to generate an electric current signal and supplies it to an RF

(Radio Frequency) amplifier 113. It should be noted that a position to radiate the laser beam is set to a predetermined position by a servo signal supplied from the servo control portion 114 to the pickup portion 112.

[0183]

Based on the electric current signal from the pickup portion 112, the RF amplifier 113 generates a focus error signal, a tracking error signal, and a reproduction signal, and supplies the tracking error signal and the focus error signal to the servo control portion 114 and supplies the reproduction signal to the signal process portion 115.

[0184]

The servo control portion 114 controls focus servo operations and tracking servo operations. Specifically, the servo control portion 114 generates a focus servo signal and a tracking servo signal based on the focus error signal and the tracking error signal from the RF amplifier 113 and supplies the generated signals to an actuator (not shown) of the pickup portion 112. The servo control portion 114 generates a spindle motor drive signal to drive the spindle motor 111 and controls a spindle servo operation to rotate the optical disk 7 at a specified rotation speed.

[0185]

Further, the servo control portion 114 provides thread control to change a position to radiate the laser beam by moving the pickup portion 112 along the radial direction of the optical disk 7. The control portion 119 settles positions to read signals on the optical disk 7. The control portion 119 controls positions of the pickup portion 112 so as to be able to read signals from settled read positions.

[0186]

The signal process portion 115 modulates recorded data supplied from a memory controller 116 to generate a recording signal and supplies it to the pickup portion 112. The signal process portion 115 also demodulates a reproduction signal from the RF amplifier 113 to generate reproduction data and supplies to the memory controller 116.

[0187]

The memory controller 116 accordingly stores recording data from the data conversion portion 118 in memory 117 as will be described later, reads the stored data, and supplies it to the signal process portion 115. Further, the memory controller 116 stores reproduction data from the signal process portion 115 in the memory 117 as appropriate, reads the stored data, and supplies it to the data conversion portion 118.

[0188]

The data conversion portion 118 creates, from an AV independent format file from the format conversion portion 12, a file of low resolution data, i.e., data resulting from reducing the amount of data placed in the AV independent format file, and supplies the AV independent format file and the low resolution data file to the memory controller 116.

[0189]

The data conversion portion 118 also supplies the format conversion portion 12 with reproduction data supplied from the memory controller 116.

[0190]

Based on operation signals and the like from an operation portion 120, the control portion 119 controls the servo control portion 114, the signal process portion 115, the memory controller 116, and the data conversion portion 118 to perform recording and reproducing processes.

[0191]

The operation portion 120 is operated by a user, for example, and supplies the control portion 119 with an operation signal corresponding to the operation.

[0192]

On the disk drive apparatus 11 configured as described above, when a user operates the operation portion 120 to instruct to record data, data supplied from the format conversion portion 12 passes through the data conversion portion 118, the memory controller 116, the signal process portion 115, and the pickup portion 112, and then is supplied to and recorded on the optical disk 7.

[0193]

Further, when a user operates the operation portion 120 to instruct to reproduce data, the data is read and reproduced from the optical disk 7 and passes through the pickup portion 112, the RF amplifier 113, the signal process portion 115, the memory controller 116, and the data conversion portion 118, and then is supplied to the format conversion portion 12.

[0194]

FIG. 18 shows an exemplary configuration of the data conversion portion 118 in FIG. 17.

[0195]

When data is recorded on the optical disk 7, the format conversion portion 12 supplies a data amount detection portion 141 with AV independent format files to be recorded composed of a video file, an audio file, and a metadata file.

[0196]

The data amount detection portion 141 unchangedly supplies the video file, the audio file, and the metadata file supplied from the format conversion portion 12 to the memory controller 116. The data amount detection portion 141 detects data amounts of the video file and the audio file and supplies the data amounts to the memory controller 116. That is, the data amount detection portion 141 detects the data amount for a specified reproduction time duration, for example, with respect to each of the video file and the audio file supplied from the format conversion portion 12 and supplies it to the memory controller 116.

[0197]

The low resolution data generation portion 142 generates a data series of the low resolution data obtained by reducing the amount of the data supplied thereto and supplies it to the memory controller 116. In this case, the low resolution data generation portion 142 outputs a file form of low resolution data. Further, the low resolution data generation portion 142 detects the data amount of low resolution data for a specified reproduction time duration, for example, and supplies it to the memory controller 116.

[0198]

In the following description, a file form of low resolution data is also referred to as a low resolution data file.

[0199]

The video file and the audio file supplied to the memory controller 116 are then supplied to and recorded on the optical disk 7 as mentioned above.

[0200]

Here, the data series of the video file and the audio file supplied from the format conversion portion 12 and the data series of low resolution data supplied from The low resolution correspond to video and audio of the same content. The video file and the audio file supplied from the format conversion portion 12 are originally to be provided for users. For this reason, video files and audio files supplied from the format conversion portion 12 are hereafter referred to as mainline data as appropriate.

[0201]

As mentioned above, the low resolution data contains video and audio data with the same content as the mainline data but has the small data amount. Accordingly, when the reproduction is performed for a specified time duration, the low resolution data can be read from the optical disk 7 faster than the mainline data.

[0202]

A data rate for the mainline data may be approximately 25 Mbps (Mega bits per second), for example. In this case, a data rate for the low resolution data may be approximately 3 Mbps, for example. Further, a data rate for the metadata may be approximately 2 Mbps, for example. In this case, a data rate for the entire data to be recorded on the optical disk 7 may is approximately 30 (= 25+3+2) Mbps. Accordingly, the optical disk 7 (or the disk drive apparatus 11 to drive it) can provide a sufficiently practicable recording rate of 35 Mbps, for example.

[0203]

As mentioned above, the data conversion portion 118 in FIG. 16 supplies the memory controller 116 with not only a data series of the mainline data (video files and audio files), but also a data series of the metadata and the low resolution data. The mainline data, the metadata, and the low resolution data supplied to the memory controller 116 are supplied to and recorded on the optical disk 7.

[0204]

On the other hand, during reproduction of data from the optical disk 7, video files, audio files, metadata files, and low resolution data files are read from the optical disk 7 and are supplied to the format conversion portion 12.

[0205]

The structure of a low resolution data file will be described with reference to FIGS. 19 through 24. As shown in FIG. 19, in a low resolution data file header, a run-in, a header partition pack, header metadata, an index table, and a filler are placed. In a low resolution data file body, essence containers separated by body partition packs are placed.

[0206]

In a low resolution data file footer, a footer partition pack and header metadata are placed. The header metadata in the footer is optional.

[0207]

The header partition pack, the header metadata, the index table, and the footer partition pack in the low resolution data file are the same as those in a standard AV multiplexing format file and a description thereof is omitted.

[0208]

The low resolution data file header is sized to be an integral multiple of the ECC block length on the optical disk 7. A set of the body partition pack and the essence container in the low resolution data file body is sized to an integral multiple of the ECC block length for the optical disk 7. The low resolution data file footer is sized to an integral multiple of the ECC block length for the optical disk 7.

[0209]

FIG. 20 is a diagram showing a configuration of the body partition pack and the essence container stored in the low resolution data file body. The essence container stores a system item, a picture essence (video data) according to the MPEG4 elementary stream, and a sound essence (sound data). A fill item is placed between the system item and the picture essence. The body partition pack, the system item, the fill item, and the picture essence are sized to be an integral multiple of the ECC block length for the optical disk 7.

[0210]

The sound essence is divided into four portions each of which is added with a filler. A set of the divided sound essence and the added filler provides the data amount equivalent to be one-half the ECC block length for the optical disk 7. That is, two sets of the divided sound essence and the filler provide the data amount equal to the ECC block length for the optical disk 7. Accordingly, the total of the sound essence and the added fillers in one essence container provides the data amount double as large as the ECC block length for the optical disk 7.

[0211]

FIG. 21 is a diagram showing a configuration of the system item and the fill item in the header. The system item stores package metadata. The fill item is composed of a filler having the KLV structure.

[0212]

FIG. 22 is a diagram showing a configuration of the picture essence. The picture essence is KLV-structured video data according to the MPEG4 elementary stream. That is, in a case of video data having the total number of scanning lines per frame rate of 525/60 (59.94), video data that is a progressive scan image at the frame rate of 24 (23.97), or video data that is a progressive scan image at the frame rate of 60 (59.94), the picture essence for one edit unit contains six GOVs (Group of VideoObjectPlane). On the other hand, in a case of video data having the total number of scanning lines per frame rate of 625/50, the picture essence for one edit unit contains five GOVs. One GOV contains intra-frame coded I-VOP (Intra Video Object Plane) at the beginning followed by the predetermined number of inter-frame forward predictive-coded P-VOPs (Predicted Video Object Plane).

[0213]

FIG. 23 is a diagram showing data amounts in a KLV-structured picture essence. In a case of video data having the total number of scanning lines per frame rate of 525/60 (59.94), the value's data amount is 384000 bytes (octets). That is, one GOV has the data amount of 6400 bytes. In this case, one GOV stores ten frames of images.

[0214]

In a case of video data that is a progressive scan image at the frame rate of 24 (23.97), the value's data amount is 384000 bytes. That is, one GOV has the data amount of 6400 bytes. In this case, one GOV stores eight frames of images.

[0215]

In a case of video data that is a progressive scan image at the frame rate of 60 (59.94), the value's data amount is 384000 bytes. That is, one GOV has the data amount of 6400 bytes. In this case, one GOV stores 20 frames of images.

[0216]

In a case of video data having the total number of scanning lines per frame rate of 625/50, the value's data amount is 384000 bytes. That is, one GOV has the data amount of 76800 bytes. In this case, one GOV stores ten frames of images.

[0217]

FIG. 24 is a diagram showing a configuration of a sound essence. The low resolution data file's sound essence is 2-channel data based on the ITU-T (International Telecommunication Union, Telecommunication Standardization Sector) G.711 standard. The sound essence is divided into four portions each of which is KLV-structured. Each of the KLV-structured divided data is added with a filler.

[0218]

In the value, samples of 2-channel are alternately placed. In a case of the sound essence corresponding to a picture essence having the total number of scanning lines per frame rate of 525/60 (59.94) the sound essence corresponding to a picture essence that is a progressive scan image at the frame rate of 24 (23.97), or the sound essence corresponding to a picture essence that is a progressive scan image at the frame rate of 60 (59.94), 16016 samples are placed in one of the four divided sound essences. On the other hand, in a case of the sound essence corresponding to a picture essence having the total number of scanning lines per frame rate of 625/50, 16000 samples are placed in one of the four divided sound essences.

[0219]

FIG. 25 is a block diagram showing a configuration of the low resolution data generation portion 142.

[0220]

The buffer 161 temporarily stores AV independent format files (master file, file-based metadata file, frame-based metadata file, auxiliary file, video file, and audio files for eight channels) supplied from the format conversion portion 12.

[0221]

A file acquisition portion 162 references the master file stored in the buffer 161 to identify file names of the file-based metadata file, the frame-based metadata file, the auxiliary file, the video file, and the audio files for eight channels. Based on the file names, the file acquisition portion 162 acquires the file-based metadata file, the frame-based metadata file, the auxiliary file, the video file, and the audio files for eight channels from the format conversion portion 12 via the buffer 161. Of these acquired files, the file acquisition portion 102 supplies the file-based metadata file and the frame-based metadata file to the metadata file process portion 163, the video file to the video file process portion 164, and the audio files for eight channels to the audio file process portion 165.

[0222]

The metadata file process portion 163 extracts file-based metadata from the file-based metadata file supplied from the file acquisition portion 162, extracts the system item in which frame-based metadata is placed from the frame-based metadata file, and supplies the file-based metadata and the system item to the data synthesis portion 166.

[0223]

The video file process portion 164 extracts the picture item from the video file supplied from the file acquisition portion 162, generates a picture essence for the low resolution data file from the extracted picture item, and supplies the picture essence to the data synthesis portion 166.

[0224]

The audio file process portion 165 extracts each channel's audio data from the audio files supplied from the file acquisition portion 162, generates low bit-rate audio data from the extracted picture item, constructs a sound essence by multiplexing and storing each channel's audio data, and supplies the sound essence to the data synthesis portion 166.

[0225]

The data synthesis portion 166 constructs a low resolution data file using the file-based metadata and the system item supplied from the metadata file process portion 163, the picture essence supplied from the video file process portion 164, and the sound essence supplied from the audio file process portion 165 and supplies the low resolution data file to a buffer 167.

[0226]

The buffer 167 temporarily stores the low resolution data file supplied from the data synthesis portion 166 and supplies the low resolution data file to the memory controller 116.

[0227]

FIG. 26 is a block diagram illustrating a configuration of the video file process portion 164. A decomposition portion 181 decomposes the video file supplied from the file acquisition portion 162 into picture items and supplies the decomposed picture items to a data conversion portion 182. The data conversion portion 182 converts the decomposed picture items into MPEG4 image data and supplies it to a KLV encoder 183. The KLV encoder 183 KLV-encodes the picture essence supplied from the data conversion portion 182 and supplies the KLV-structured picture essence to the data synthesis portion 166.

[0228]

FIG. 27 is a block diagram illustrating a configuration of the audio file process portion 165. A KLV decoder 201 decomposes the KLV structure of the body in the each channel's audio file supplied from the file acquisition portion 162 and supplies a data conversion portion 202 with resulting WAVE-formatted audio data for each channel.

[0229]

The data conversion portion 202 converts the each channel's WAVE-formatted audio data supplied from the KLV decoder 201 into 2-channel audio data according to the ITU-T G.711 format and supplies the audio data to a channel multiplexing portion 203. The channel multiplexing portion 203 multiplexes the 2-channel audio data supplied from the data conversion portion 202 in units of samples and supplies resulting multiplexed audio data to the KLV encoder 204. The KLV encoder 204 divides the audio data supplied from the channel multiplexing portion 203 into four portions, KLV-codes each divided audio data according to the KLV structure, and supplies the KLV-structured audio data to a filler generation portion 205.

[0230]

The filler generation portion 205 generates a filler for each KLV-structured audio data, adds the filler to the audio data, and supplies it to a KLV encoder 206. The KLV encoder 206 KLV-codes the filler added to the audio data according to the KLV structure and outputs the sound essence added with the KLV-structured filler.

[0231]

FIG. 28 is a block diagram showing a configuration of the data synthesis portion 166. A multiplexing portion 221 multiplexes the system item supplied from the metadata file process portion 163, the video essence supplied from the video file process portion 164, and the sound essence supplied from the audio file process portion 165, adds a body partition to generate a body, and supplies the generated body to a footer generation portion 222. The footer generation portion 222 generates a footer, adds the footer to the body, and supplies the body and the footer to a header generation portion 223.

[0232]

The header generation portion 223 generates a header, adds the header to the body and the footer, and supplies the body, the footer, and the header to a filler generation portion 224. The filler generation portion 224 generates a filler to be added to the header, adds the generated filler to the header, and outputs a low resolution file added with the filler.

[0233]

FIG. 29 is a flowchart showing a video file process. At Step S101, the file acquisition portion 162 acquires the video file body from the format conversion portion 12 via the buffer 161, and the process proceeds to Step S102. At Step S102, the decomposition portion 181 decomposes the video file supplied from the file acquisition portion 162 into picture items, and the process proceeds to Step S103. At Step S103, the data conversion portion 182 converts the decomposed picture item into MPEG image data, and the process proceeds to Step S104. At Step S104, the KLV encoder 183 KLV-encodes the picture item supplied from the data conversion portion 182 according to the KLV structure to compose a picture essence. The video file process then terminates.

[0234]

FIG. 30 is a flowchart showing an audio file process. At Step S121, the file acquisition portion 162 acquires the audio file body from the format conversion portion 12 via the buffer 161, and the process proceeds to Step S122. At Step S122, the KLV decoder 201 decomposes the KLV structure of the audio file body in each channel supplied from the file acquisition portion 162, and the process proceeds to Step S123.

[0235]

At Step S123, the data conversion portion 202 converts each channel's audio data of the WAVE format supplied from the KLV decoder 201 into 2-channel audio data of the ITU-T G.711 format, and the process proceeds to Step S124. At Step S124, the channel multiplexing portion 203 multiplexes the 2-channel audio data supplied from the data conversion portion 202 in units of samples, and the process proceeds to Step S125. At Step S125, the KLV encoder 204 divides the audio data supplied from the channel multiplexing portion 203 into four portions and KLV-codes each divided audio data according to the KLV structure, and the process proceeds to Step S126.

[0236]

At Step S126, the filler generation portion 205 generates a filler for each KLV-structured audio data and adds the filler to the audio data. The process proceeds to Step S127. At Step S127, the KLV encoder 206 KLV-codes the filler added to the audio data according to the KLV structure to compose a sound essence. The audio file process then terminates.

[0237]

FIG. 31 is a flowchart illustrating a metadata file process. At Step S141, the file acquisition portion 162 acquires the metadata file body from the format conversion portion 12 via the buffer 161 to compose a system item, and the process proceeds to Step S142. At Step S142, the metadata file process portion 163 generates a filler, and the process proceeds to Step S143. At Step S143, the metadata file process portion 163 KLV-codes the filler added to the system item according to the KLV structure to compose a fill item and outputs the system item added with the fill item. The metadata file process then terminates.

[0238]

FIG. 32 is a flowchart illustrating a low resolution data file synthesis process. At Step S161, the multiplexing portion 221 multiplexes the system item supplied from the metadata file process portion 163, the video essence supplied from the video file process portion 164, and the sound essence supplied from the audio file process portion 165 to generate an essence container, and the process proceeds to Step S162. At Step S162, the multiplexing portion 221 adds a body partition to the essence container to generate a body, and the process proceeds to Step S163.

[0239]

At Step S163, the data synthesis portion 166 outputs the body, and the process proceeds to Step S164. At Step S164, the footer generation portion 222 generates a footer, and the process proceeds to Step S165. At Step S165, the data synthesis portion 166 outputs the footer, and the process proceeds to Step S166.

[0240]

At Step S166, the header generation portion 223 generates a header, and the process proceeds to Step S167. At Step S167, the filler generation portion 224 generates a filler to be added to the header, and the process proceeds to Step S168. At Step S168, the data synthesis portion 166 outputs the header added with the filler. The low resolution data file synthesis process then terminates.

[0241]

Next, referring to the flowchart in FIG. 33, the recording process performed by the control portion 119 will be described.

[0242]

When an operation signal to start a recording process in supplied fron the operation portion 120 to the control portion 119 upon an operation on the operation portion 120, the control portion 119 starts the recording process.

[0243]

At Step S231, the control portion 119 first configures, in addition to audio tree ring size Tsa and video tree ring size Tsv, low resolution tree ring size Tsl and meta tree ring size Tsm.

[0244]

Audio tree ring size Tsa is a variable to determine the data amount of audio files collectively arranged and recorded on the optical disk 7. Audio tree ring size Tsa is represented by the reproduction time of an audio file, for example. Similarly, video tree ring size Tsv is a variable to determine the data amount of video files collectively arranged and recorded on the optical disk 7. Video tree ring size Tsv is represented by the reproduction time of a video file, for example.

[0245]

It should be noted that the reason why audio tree ring size Tsa and video tree ring size Tsv are, so to speak, indirectly represented in reproduction times, not in the data amount itself such as the number of bits or bytes is as follows.

[0246]

As will be described later, the recording process in FIG. 33 cyclically arranges and records on the optical disk 7, audio tree ring data that is a collection of audio files in units of data amounts based on audio tree ring size Tsa extracted from a series of audio files and video tree ring data that is a collection of video files in units of data amounts based on video tree ring size Tsv extracted from a series of video files.

[0247]

As mentioned above, in a case where the audio tree ring data and the video tree ring data are cyclically arranged and recorded on the optical disk 7, in consideration of audiovisual reproduction, the reproduction is not performed until there are both an available video file and an audio file attendant thereon. From the viewpoint of such reproduction, it is necessary to record audio tree ring data for a given reproduction time slot and tree ring data for that reproduction time slot at close positions, e.g., adjacently to each other, on the optical disk 7.

[0248]

Generally, however, there is a large difference between the data amount of the audio file and that of the video file for the same reproduction time period. That is, the data amount of the audio file for a given reproduction time is considerably smaller than that of the video file for that reproduction time period. Further, there may be a case where the audio file or the video file may use a variable data rate, not a fixed one.

[0249]

Accordingly, when audio tree ring size Tsa and video tree ring size Tsv are expressed in data amounts and the audio tree ring data and the video tree ring data are sequentially extracted in units of the data amounts from the series of audio files and video files, respectively, for each piece of video tree ring data for each reproduction time slot, audio tree ring data for a later (advanced) reproduction time slot corresponding to the gradually advancing reproduction time. As a result, it becomes difficult to arrange the audio file and the video file to be reproduced for the same reproduction time slot at close positions on the optical disk 7.

[0250]

On the other hand, when audio tree ring size Tsa and video tree ring size Tsv are expressed in reproduction times and the audio tree ring data and the video tree ring data for the reproduction time are extracted in units of data amounts sequentially from series of audio files and video files, respectively, it is possible to acquire a set of audio tree ring data and video tree ring data for similar reproduction time slots. As a result, it is possible to arrange audio files and video files to be reproduced for the same reproduction time slot at close positions.

[0251]

Here, it is desirable to set audio tree ring size Tsa to a value so that it makes seeking and skipping faster than reading, from the optical disk 7, audio tree ring data having the data amount equivalent to the reproduction time expressed by audio tree ring size Tsa. The same also applies to video tree ring size Tsv. According to the experience of the inventors, such video tree ring size Tsv is 1.5 to 2 seconds, for example.

[0252]

In addition, when configuring audio tree ring data and video tree ring data for similar reproduction time slots, it is only necessary to assign the same value to audio tree ring size Tsa and video tree ring size Tsv. In this case, it is desirable to alternately arrange audio tree ring data and video tree ring data for similar reproduction time slots on the optical disk 7 from the viewpoint of the reproduction as mentioned above.

[0253]

Further, audio tree ring size Tsa and video tree ring size Tsv can be different values, and in consideration that an audio file's data rate is generally considerably lower than a video file's data rate, audio tree ring size Tsa can be twice as large as video tree ring size Tsv, for example. In this case, a piece of audio tree ring data corresponds to two pieces of video tree ring data for a reproduction time slot similar to the reproduction time slot of the audio tree ring data. From the viewpoint of the reproduction as mentioned above, it is desirable to arrange one piece of audio tree ring data and two pieces of corresponding video tree ring data at close positions on the optical disk 7. Specifically, it is desirable to cyclically arrange one piece of audio tree ring data and two pieces of corresponding video tree ring data in the order of: audio tree ring data and two pieces of corresponding video tree ring data one after another; or one of two pieces of corresponding video tree ring data, audio tree ring data, and the other piece of video tree ring data, for example.

[0254]

At Step S1, audio tree ring size Tsa and video tree ring size Tsv may be set to predetermined fixed values or variable values. When audio tree ring size Tsa and video tree ring size Tsv are set to variable values, the variable values can be input by operating the operation portion 120, for example.

[0255]

In addition, low resolution tree ring size Tsl is a variable to determine the data amount of low resolution data collectively arranged and recorded on the optical disk 7, and is expressed in the reproduction time for the video file (or the audio file) originating the low resolution data, similarly to audio tree ring size Tsa and video tree ring size Tsv as mentioned above. Similarly, meta tree ring size Ts is a variable to determine the data amount of metadata collectively arranged and recorded on the optical disk 7, and is expressed in the reproduction time of a video file (or an audio file) whose metadata describes various information (e.g., the date and time to capture an image), similarly to audio tree ring size Tsa and video tree ring size Tsv as mentioned above, for example.

[0256]

It should be noted that the reason why low resolution tree ring size Tsl and meta tree ring size Tsm are, so to speak, indirectly represented in reproduction times, not in the data amount itself such as the number of bits or bytes is the same as that for audio tree ring size Tsa and video tree ring size Tsv as mentioned above.

[0257]

As will be described later, the recording process in FIG. 33 cyclically arranges and records, not only audio tree ring data that is a collection of audio files in units of data amounts based on audio tree ring size Tsa extracted from a series of audio files and video tree ring data that is a collection of video files in units of data amounts based on video tree ring size Tsv extracted from a series of video files, but also low resolution tree ring data that is a collection of low resolution data in units of data amounts based on low resolution tree ring size Tsl extracted from a data series of low resolution data and meta tree ring data that is a collection of metadata in units of data amounts based on meta tree ring size Tsm extracted from a data series of metadata.

[0258]

As described above, in a case where audio tree ring dataand video tree ring data, low resolution tree ring data, and meta tree ring data are cyclically arranged on the optical disk 7, because the low resolution tree ring data is obtained by reducing the data amount of the audio tree ring data or the video tree ring data, udio tree ring data and video tree ring data for a given reproduction time slot and low resolution tree ring data, i.e., an equivalence to the audio tree ring data or the video tree ring data with the reduced data amount for that reproduction time slot, should be arranged at close positions on the optical disk 7. Further, because the meta tree ring data represents information about the audio tree ring data or the video tree ring data, audio tree ring data and video tree ring data for a given reproduction time slot and meta tree ring data representing information about the audio tree ring data or the video tree ring data for that reproduction time slot should be recorded at close positions on the optical disk 7.

[0259]

However, when comparing the data rate of the audio file or the video file with that of the low resolution data or the metadata for the same reproduction time period, the data rate of the low resolution data or the metadata is smaller than that of the audio file or the video file.

[0260]

Accordingly, representing low resolution tree ring size Tsl and meta tree ring size Tsm in data amounts causes, a problem similar to that of representing audio tree ring size Tsa and video tree ring size Tsv in data amounts, that is, it becomes difficult to arrange the audio file, the video file, the low resolution data, and the metadata to be reproduced for similar reproduction time slots at close positions on the optical disk 7.

[0261]

In this regard, the embodiment in FIG. 33 uses reproduction times to represent low resolution tree ring size Tsl and meta tree ring size Tsm as well as audio tree ring size Tsa and video tree ring size video tree ring size Tsv. This makes it possible to arrange the audio file, the video file, the low resolution data, and the metadata to be reproduced for similar reproduction time slots at close positions on the optical disk 7.

[0262]

It should be noted that, at Step S231, predetermined fixed values or variable values may be used for audio tree ring size Tsa, video tree ring size Tsv, low resolution tree ring size Tsl, and meta tree ring size Tsm. When audio tree ring size Tsa, video tree ring size Tsv, low resolution tree ring size Tsl, and meta tree ring size Tsm are set to variable values, the variable values can be input by operating the operation portion 120, for example.

[0263]

After Step S231, the process proceeds to Step S232, and the control portion 110 starts a low resolution data generation process to generate a low resolution data series from the audio file and the video file supplied from the format conversion portion 12 to the disk drive apparatus 11, and controls the memory controller 116 to start an audio file storage process and a video file storage process to supply the audio file and the video file obtained in the data conversion portion 118 to the memory 117 for storing these files. Further, at Step S232, the control portion 119 controls the memory controller 116 to start a metadata storage process and a low resolution data storage process to supply the metadata and the low resolution data obtained in the data conversion portion 118 to the memory 117 for storing these data.

[0264]

The process proceeds to Steps S233 and S234, and the control portion 119 starts an audio file recording task that is a control task to record an audio file on the optical disk 7, and at Step S234, starts a video file recording task that is a control task to record a video file on the optical disk 7. The process proceeds to Step S235. At Step S235, the control portion 119 starts a low resolution data recording task that is a control task to record low resolution data on the optical disk 7, and the process proceeds to Step S236. At Step S236, the control portion 119 starts a metadata recording task that is a control task to record metadata on the optical disk 7, and the process proceeds to Step S237. It should be noted that detailed descriptions will be provided later for the audio file recording task at Step S233, the video file recording task at Step S234, the low resolution data recording task at Step S235, and the metadata recording task at Step S236.

[0265]

At Step S237, the control portion 119 determines whether or not the operation portion 120 has supplied an operation signal to instruct the termination of data recording. When it is determined that no such operation signal is supplied, the process proceeds to Step S238, and the control portion 119 determines whether or not all recording tasks have terminated. When it is determined at Step S238 that all recording tasks have not terminated, the process returns to Step S237. The similar process is repeated thereafter.

[0266]

At Step S238, if it is determined that all recording tasks have been terminated, that is, if all of the audio file recording task initiated at Step S233, the video file recording task initiated at Step S234, the low resolution data recording task initiated at Step S235, and the metadata recording task initiated at Step S236 have terminated, the recording process terminates.

[0267]

At Step S237, on the other hand, if it is determined that an operation signal to instruct the termination of data recording is supplied, that is, if a user, for example, operates the operation portion 120 to terminate the data recording, the process proceeds to Step S239, and the control portion 119 terminates the low resolution data generation process, the audio file storage process, the video file storage process, the metadata storage process, and the low resolution data storage process initiated at Step S232. The process proceeds to Step S240.

[0268]

Step S240, similarly to at Step S238, it is determined whether or not all the recording tasks have terminated. When it is determined at Step S240 that all the recording tasks have not terminated, the process returns to Step S240 and waits until all the recording tasks terminate.

[0269]

Further, at Step S240, if it is determined that all the recording tasks have terminated, that is, if all of the audio file recording task initiated at Step S233, the video file recording task initiated at Step S234, the low resolution data recording task initiated at Step S235, and the metadata recording task initiated at Step S236 have terminated, the recording process terminates.

[0270]

Next, referring to the flowchart in FIG. 34, the audio file recording task initiated at Step S233 in FIG. 33 will be described.

[0271]

When the audio file recording task starts, the control portion 119 first initializes variable Na to, for example, 1, at Step S251. Variable Na is incremented by one in the process at Step S257 later. The process proceeds to Step S252.

[0272]

At Step S252, similarly to at Step S12 in FIG. 5, the control portion 119 determines whether or not Tsa x Na is smaller than or equal to Tsv x Nv. Further, the control portion 119 determines whether or not Tsa x Na is smaller than or equal to Tsl x Nl and is smaller than or equal to Tsm x Nm.

[0273]

Here, Tsa denotes an audio tree ring size representing a given reproduction time for an audio file. As will be described later, variable Na is incremented by one each time an audio file (audio tree ring data) having the data amount based on audio tree ring size Tsa is recorded on the optical disk 7. Similarly, Tsv denotes a video tree ring size video tree ring size, and as will be described later, variable Nv is incremented by one each time the video file recording task records a video file (video tree ring data) having the data amount based on video tree ring size Tsv on the optical disk 7. Accordingly, Tsa x Na is equivalent to the last reproduction time for audio tree ring data to be recorded on the optical disk 7 when the audio file is recorded in units of audio tree ring size Tsa. Tsv x Nv is equivalent to the last reproduction time for video tree ring data to be recorded on the optical disk 7 when the video file is recorded in units of video tree ring size video tree ring size Tsv.

[0274]

Tsl denotes a low resolution tree ring size, and as will be described later, variable Nl is incremented by one each time the low resolution data recording task records low resolution data (low resolution tree ring data) having the data amount based on low resolution tree ring size Tse on the optical disk 7. Further, Tsm denotes a meta tree ring size, and as will be described later, variable Nm is incremented by one each time the metadata recording task records metadata (meta tree ring data) having the data amount based on meta tree ring size Tsm on the optical disk 7. Accordingly, Tsl x Nl is equivalent to the last reproduction time for the low resolution tree ring data to be recorded on the optical disk 7 when the low resolution data is recorded in units of low resolution tree ring size Tse. Tsm x Nm is equivalent to the last reproduction time for the meta tree ring data to be recorded on the optical disk 7 when the metadata is recorded in units of meta tree ring size Tsm.

[0275]

On the other hand, let us suppose to cyclically arrange audio tree ring data, video tree ring data, low resolution tree ring data, and meta tree ring data so that they are recorded at close positions on the optical disk 7 in terms of similar reproduction time slots. Further, let us suppose to arrange audio tree ring data, video tree ring data, low resolution tree ring data, and meta tree ring data so that they are arranged at earlier positions on the optical disk 7 (at positions toward the beginning of a sequence of reading or writing data on the optical disk 7) correspondingly to earlier reproduction times. Moreover, let us suppose to arrange audio tree ring data, video tree ring data, low resolution tree ring data, and meta tree ring data for similar reproduction time slots in the order of audio tree ring data, video tree ring data, low resolution tree ring data, and meta tree ring data, for example, at earlier positions on the optical disk 7.

[0276]

In this case, targeted audio tree ring data, that is, audio tree ring data to be recorded next, corresponds to the most recent reproduction time slot (closest to reproduction time Tsa x Na) prior to reproduction time Tsa x Na. The targeted audio tree ring data needs to be recorded immediately before video tree ring data, low resolution tree ring data, and meta tree ring data recorded for the most recent reproduction time slot earlier than reproduction time Tsa x Na, that is, the targeted audio tree ring data needs to be recorded immediately after video tree ring data, low resolution tree ring data, and meta tree ring data recorded for the second most recent reproduction time slot earlier than reproduction time Tsa x Na.

[0277]

Incidentally, video tree ring data to be recorded corresponds to the most recent reproduction time slot earlier than Tsv x Nv. In addition, low resolution tree ring data to be recorded corresponds to the most recent reproduction time slot earlier than Tsl x Nl, and meta tree ring data to be recorded corresponds to the most recent reproduction time slot earlier than Tsm x Nm. When tree ring data belongs to similar reproduction time slots, since the audio tree ring data is arranged at earlier positions on the optical disk 7 as mentioned above, the targeted audio tree ring data needs to be recorded at such timing when reproduction time Tsa x Na for the audio tree ring data is smaller than or equal to reproduction time Tsv x Nv for the video tree ring data, reproduction time Tsl x Nl for the low resolution tree ring data, and reproduction time Tsm x Nm for the meta tree ring data.

[0278]

In this regard, at Step S252, as mentioned above, it is determined whether or not reproduction time Tsa x Na for the audio tree ring data is smaller than or equal to reproduction time Tsv x Nv for the video tree ring data, reproduction time Tsl x Nl for the low resolution tree ring data, and reproduction time Tsm x Nm for the meta tree ring data. This determines whether or not the current timing is a timing when targeted audio tree ring data should be recorded.

[0279]

At Step S252, if it is determined that reproduction time Tsa x Na for the audio tree ring data is not smaller than or equal to (before) any of reproduction time Tsv x Nv for the video tree ring data, reproduction time Tsl x Nl for the low resolution tree ring data, and reproduction time Tsm x Nm for the meta tree ring data, that is, if the current timing is not a timing when the targeted audio tree ring data should be recorded, the process returns to Step S252, and the similar process is repeated thereafter.

[0280]

Further, at Step S252, if it is determined that reproduction time Tsa x Na for the audio tree ring data is smaller than or equal to all of reproduction time Tsv x Nv for the video tree ring data, reproduction time Tsl x Nl for the low resolution tree ring data, and reproduction time Tsm x Nm for the meta tree ring data, that is, if the current timing is a timing when the targeted audio tree ring data should be recorded, the process proceeds to Step S253, and the control portion 119 determines whether or not an audio file is supplied to the memory 117 from the data conversion portion 118 via the memory controller 116. When it is determined that the audio file is supplied, the process proceeds to Step S254.

[0281]

At Step S254, the control portion 119 determines whether or not audio files needed for the reproduction equivalent to audio tree ring size Tsa x Na in total have been stored in the memory 117. When it is determined that sufficient audio files have not yet been stored in the memory 117, the process returns to Step S252, and the succeeding process is repeated. When it is determined at Step S254 that the audio files equivalent to reproduction time Tsa x Na have been stored in the memory 117, the process proceeds to Step S255.

[0282]

It should be noted that if audio files needed for the reproduction equivalent to reproduction time Tsa x Na in total are detected, the data amount detection portion 141 of the data conversion portion 118 notifies the memory controller 116 of the detection. Based on the notification, the memory controller 116 determines whether or not audio files needed for the reproduction equivalent to reproduction time Tsa x Na in total has been stored in the memory 117, and the determination result thereof is notified to the control portion 119. That is, the control portion 119 performs the determination at step S254 based on the determination result from the memory controller 116.

[0283]

Here, FIG. 35 shows the relationship between the total data amount (total data amount) La of audio files to be stored in the memory 117 and the time (reproduction time). On the right of FIG. 6, a small vertical bidirectional arrow (an arrow indicating an interval between horizontal dotted lines) represents data amount Bu of an ECC block. Dotted line Lv in FIG. 35 indicates total data amount (total data amount) Lv of video files to be stored in the memory 117 and corresponds to a solid line in FIG. 9 to be described later. Further in FIG. 35, a straight line is used to represent the audio file's total data amount La. This indicates that the audio file's data rate is fixed. However, variable data rates may be used for audio files.

[0284]

In FIG. 35, if Na = 1, for example, then the data amount of audio file needed for the reproduction equivalent to time Tsa x Na (=1) is AN1'. Accordingly, at Step S254 where Na = 1, when an audio file with the total data amount of AN1' is stored in the memory 117, it is determined that the audio file equivalent to reproduction time Tsa x Na is stored in the memory 117, and the process proceeds to Step S255.

[0285]

At Step S255, the control portion 119 controls the memory controller 116 to extract, from audio files stored in the memory 117, an audio file with the data amount that is an integral multiple of (multiplied by n) data amount Bu of one ECC block, for example, as a unit of reading or writing to the optical disk 7, and is equivalent to a maximum data amount capable of being read from the memory 117, by reading the audio files in the chronological order of input. The process proceeds to Step S256. It should be noted that the audio tree ring data to be read from the memory 117 as the audio file whose data amount is an integral multiple of the ECC block and is equivalent to a maximum data amount capable of being read from the memory 117 corresponds to the most recent audio tree ring data before reproduction time Tsa x Na as mentioned above.

[0286]

Here, when the time is 1 x Tsa in FIG. 35 described above, the memory 117 stores an audio file whose the data amount is at least AN1'. Since data amount AN1' is greater than the data amount of one ECC block and is smaller than the data amount of two ECC blocks, at Step S255, an audio file with data amount AN1 equal to data amount Bu of one ECC block is read from the memory 117 as targeted audio tree ring data and thereby extracted.

[0287]

If there is an audio file that is not read at Stėp S255, that is, at the time 1 x Tsa in FIG. 35, an audio file whose data amount Aa1 is smaller than data amount Bu of one ECC block is left in the memory 117 as it is.

[0288]

Returning to FIG. 34, at Step S256, the control portion 119 causes the memory controller 116 to supply the signal process portion 115 with targeted audio tree ring data obtained at Step S255 having the data amount equivalent to an integral multiple of the ECC block. Accordingly, recording control is performed so as to record the targeted audio tree ring data having the data amount equivalent to an integral multiple of the ECC block onto as many ECC blocks as the integral multiple.

[0289]

At the time 1 x Tsa in FIG. 35, an audio file equivalent to data amount Bu of one ECC block is supplied as targeted audio tree ring data from the memory controller 116 to the signal process portion 115. Then, the targeted audio tree ring data having data amount Bu of one ECC block is supplied to the pickup portion 112, and as shown in FIG. 36, recorded onto ECC block #1 that is one of ECC blocks on the optical disk 7, so that boundaries of the audio tree ring data match those of the ECC block #1 on the optical disk 7.

[0290]

Here, for simplicity, the optical disk 7 is assumed to contain a physically contiguous, sufficiently large free area. Further, when data is read or written to the optical disk 7 from inside peripheries to outside peripheries, it is assumed that data is continuously recorded on free areas from inside peripheries to outside peripheries in the order of data supplied from the memory controller 116 to the signal process portion 115.

[0291]

After the recording control for the targeted audio tree ring data is performed at Step S256 as mentioned above, the process proceeds to Step S257, and the control portion 119 increments variable Na by 1. The process returns to Step S252, and the succeeding process is performed.

[0292]

Meanwhile, at Step S253, if it is determined that no audio file is supplied to the memory 117, that is, the data conversion portion 118 stops supplying an audio file to the memory controller 116, the process proceeds to Step S258, and the control portion 119 controls the memory controller 116 to read all audio files remaining in the memory 117, and allows the memory controller 116 to supply the corresponding audio tree ring data to the signal process portion 115. Accordingly, the recording control is performed such that the audio tree ring data having the data amount equivalent to an integral multiple of the ECC block is recorded onto as many ECC blocks as the integral multiple.

[0293]

As mentioned above, an audio file has the data amount equivalent to an integral multiple of the ECC block. Thus, at Step S253, the audio tree ring data having the data amount equivalent to an integral multiple of the ECC block is recorded onto as many ECC blocks as the integral multiple.

[0294]

Thereafter, the process proceeds to Step S259, and the control portion 119 sets variable Na to a value equivalent to the infinity (a very large value). The audio file recording task then terminates.

[0295]

Accordingly, in the audio file recording task in FIG. 34, the audio tree ring data having the data amount, for example, equivalent to an integral multiple of the ECC block as units of reading and writing to the optical disk 7 is cyclically recorded onto as many ECC blocks as the integral multiple so that the boundaries of the audio tree ring data match those of the ECC block on the optical disk 7.

[0296]

Next, referring to the flowchart in FIG. 36, the video file recording task initiated at Step S234 in FIG. 33 will be described.

[0297]

When the video file recording task starts, the control portion 119 first initializes variable Nv to 1, for example, at Step S261. Variable Nv is incremented by one in the process at Step S267 later. The process then proceeds to Step S262.

[0298]

At Step S262, the control portion 119 determines whether or not Tsv x Nv is smaller than Tsa x Na, and equal or smaller than Tsl x Nl, and Tsm x Nm.

[0299]

In this regard, Tsa x Na is equivalent to the last reproduction time for audio tree ring data to be recorded on the optical disk 7 when audio files are recorded in units of audio tree ring size Tsa. Tsv x Nv is equivalent to the last reproduction time for video tree ring data to be recorded on the optical disk 7 when video files are recorded in units of video tree ring size Tsv.

[0300]

It is assumed that audio tree ring data and video tree ring data are cyclically arranged so that these data for similar reproduction time slots are recorded at close positions on the optical disk 7 as mentioned above. Further, it is assumed that, in terms of the arrangement of audio tree ring data and video tree ring data for similar reproduction time slots, audio tree ring data precedes video tree ring data. Video tree ring data to be recorded next is referred to as targeted video tree ring data. The targeted video tree ring data corresponds to video tree ring data for the most recent reproduction time slot (closest to reproduction time Tsv x Nv) before reproduction time Tsv x Nv. The targeted video tree ring data needs to be recorded immediately after audio tree ring data recorded for the most recent reproduction time slot before reproduction time Tsa x Na. Accordingly, the targeted video tree ring data needs to be recorded at the timing when reproduction time Tsv x Nv for the video tree ring data is smaller than reproduction time Tsa x Na for the audio tree ring data.

[0301]

In this regard, at Step S262, it is determined whether or not reproduction time Tsv x Nv for the video tree ring data is smaller than reproduction time Tsa x Na for the audio tree ring data as mentioned above. Accordingly, it is determined whether or not the current timing is for a timing when targeted video tree ring data should be recorded.

[0302]

Further, similar to the case at Step S252 in FIG. 34, the condition that Tsv x Nv is smaller than or equal to Tsl x Nl is required for recording the targeted video tree ring data that is video tree ring data to be recorded next, i.e., video tree ring data for the most recent reproduction time slot (closest to reproduction time Tsv x Nv) before reproduction time Tsv x Nv, immediately before the low resolution tree ring data for the most recent reproduction time slot before reproduction time Tsv x Nv, in other words, immediately after the low resolution tree ring data recorded for the next most recent reproduction time slot before reproduction time Tsv x Nv.

[0303]

Further, similar to the case at Step S252 in FIG. 34, the condition that Tsv x Nv is smaller than or equal to Tsm x Nm is required for recording the targeted video tree ring data that is video tree ring data to be recorded next, i.e., video tree ring data for the most recent reproduction time slot before reproduction time Tsv x Nv, immediately before the meta tree ring data for the most recent reproduction time slot before reproduction time Tsv x Nv, in other words, immediately after the meta tree ring data recorded for the next most recent reproduction time slot before reproduction time Tsv x Nv.

[0304]

At Step S262, if it is determined that reproduction time Tsv x Nv for the video tree ring data is not smaller than reproduction time Tsa x Na for the audio tree ring data, not smaller than or equal to reproduction time Tsl x Nl for the low resolution tree ring data, or not smaller than or equal to reproduction time Tsm x Nm for the meta tree ring data, that is, if the current timing is not a timing when the targeted video tree ring data should be recorded, the process returns to Step S262. The similar process is repeated thereafter.

[0305]

At Step S262, if it is determined that reproduction time Tsv x Nv for the video tree ring data is smaller than reproduction time Tsa x Na for the audio tree ring data, smaller than or equal to reproduction time Tsl x Nl for the low resolution tree ring data, and smaller than or equal to reproduction time Tsm x Nm for the meta tree ring data, that is, the current timing is a timing when the targeted video tree ring data should be recorded, the process returns to Step S263, and the control portion 119 determines whether or not a video file is supplied to the memory 117 from the data conversion portion 118 via the memory controller 116. When it is determined that the video file is supplied, the process proceeds to Step S24.

[0306]

At Step S264, the control portion 119 determines whether or not video files needed for the reproduction equivalent to video tree ring size Tsv x Nv in total have been stored in the memory 117. When it is determined that sufficient video files have not yet been stored in the memory 117, the process returns to Step S262, and the succeeding process is repeated. When it is determined at

Step S264 that the memory 117 stores video files equivalent to reproduction time Tsv x Nv, the process proceeds to Step S265.

[0307]

It should be noted that when video files needed for the reproduction equivalent to reproduction time Tsv x Nv in total are detected, he data amount detection portion 141 of the data conversion portion 118 notifies the memory controller 116 of the detection. Based on the notification, the memory controller 116 determines whether or not video files needed for the reproduction equivalent to reproduction time Tsv x Nv in total have been stored in the memory 117, and notifies the control portion 119 of the determination result. That is, the control portion 119 performs the determination at step S264 based on the determination result from the memory controller 116.

[0308]

Here, FIG. 40 shows the relationship between the total data amount (total data amount) La of video files to be stored in the memory 117 and the time (reproduction time). On the right of FIG. 40, similarly to FIG. 35, a small vertical bidirectional arrow (an arrow indicating an interval between horizontal dotted lines) represents data amount Bu of an ECC block. Dotted line La in FIG. 40 indicates total data amount La of audio files to be stored in the memory 117 and corresponds to a solid line in FIG. 35 above.

[0309]

In FIG. 40, if Nv = 1, for example, then the data amount of video file needed for the reproduction equivalent to time Tsv x Nv (=1) is VN1'. Accordingly, at Step S264 where Nv = 1, when an video file with the total data amount of VN1' is stored in the memory 117, it is determined that the video file equivalent to reproduction time Tsv x Nv is stored in the memory 117, and the process proceeds to Step S265.

[0310]

At Step S265, the control portion 119 controls the memory controller 116 to extract, from video files stored in the memory 117, an video file with the data amount that is an integral multiple of (multiplied by n) data amount Bu of one

ECC block, for example, as a unit of reading or writing to the optical disk 7, and is equivalent to a maximum data amount capable of being read from the memory 117, by reading the video files in the chronological order of input. The process proceeds to Step S266. It should be noted that the video tree ring data to be read from the memory 117 as the video file whose data amount is an integral multiple of the ECC block and is equivalent to a maximum data amount capable of being read from the memory 117 corresponds the most recent video tree ring data before reproduction time Tsv x Nv as mentioned above.

[0311]

Here, when the time is 1 x Tsv in FIG. 35 described above, the memory 117 stores an video file whose the data amount is at least VN1'. Since data amount VN1' is greater than the data amount of one ECC block and is smaller than the data amount of two ECC blocks, at Step S265, an video file with data amount VN1 equal to data amount Bu of one ECC block is read from the memory 117 as targeted video tree ring data and thereby extracted.

[0312]

If there is a video file that is not read at Step S265, that is, at the time 1 x Tsv in FIG. 40, a video file whose data amount Va1 is smaller than data amount Bu of one ECC block is left in the memory 117 as it is.

[0313]

Returning to FIG. 39, at Step S266, the control portion 119 allows the memory controller 116 to supply the signal process portion 115 with targeted video tree ring data obtained at Step S265 having the data amount equivalent to an integral multiple of the ECC block. Accordingly, recording control is performed so as to record the targeted video tree ring data having the data amount equivalent to an integral multiple of the ECC block onto as many ECC blocks as the integral multiple.

[0314]

At the time 1 x Tsv in FIG. 40, a video file equivalent to data amount Bu of four ECC blocks is supplied as targeted video tree ring data from the memory controller 116 to the signal process portion 115. Then, the targeted video tree ring data having data amount Bu of four ECC blocks is supplied to the pickup portion 112, and as shown in FIG. 36 described above, recorded onto ECC blocks #2, #3, #4, #5, that are four ECC blocks on the optical disk 7, so that boundaries of the video tree ring data match those of the ECC block #2 to #5 on the optical disk 77 (the boundary at the beginning of ECC block #2 and the boundary at the end of ECC block #5).

[0315]

In other words, for simplicity, let us assume that audio tree ring size Tsa equals video tree ring size Tsv. After the audio file recording task in FIG. 34 and the video file recording task in FIG. 39 start, under the condition of Na = Na = 1, ECC block #1 records the most recent audio tree ring data before reproduction time Tsa x Na as shown in FIG. 36. When the audio tree ring data is recorded on ECC block #1, the audio file recording task in FIG. 34 increments variable Na by one at Step S257 to set Na = 2. At this time, variable Nv remains 1, thus reproduction time Tsa x Na becomes smaller than reproduction time Tsa x Na. As a result, the video file recording task in FIG. 39 records the most recent video tree ring data before reproduction time Tsv x Nv onto ECC blocks #2 through #5 at Step S266.

[0316]

In other words, as mentioned above, because it is assumed herein that data is continuously recorded on free areas from inside peripheries to outside peripheries of the optical disk 7 in the order of data supplied from the memory controller 116 to the signal process portion 115, the video tree ring data equivalent to four ECC blocks, i.e., the most recent video tree ring data before reproduction time Tsv x Nv, begins with ECC block #2 immediately after ECC block #1 that has recorded audio tree ring data immediately before. Consequently, the video tree ring data is recorded onto ECC blocks #2 through #5 as shown in FIG. 36.

[0317]

According to the above-mentioned description, audio tree ring data and video tree ring data obtained under the condition of Na = Na = 1, that is, the most recent audio tree ring data before reproduction time Tsa x Na and the most recent video tree ring data before reproduction time Tsv x Nv equal to reproduction time Tsa x Na, in other words, audio tree ring data and video tree ring data for similar reproduction time slots are arranged and recorded at adjacent positions on the optical disk 7.

[0318]

After the control is provided to record the targeted video tree ring data as mentioned above at Step S266, the process proceeds to Step S267. The control portion 119 increments variable Nv by one. The process returns to Step S262, and the succeeding process is repeated.

[0319]

Meanwhile, at Step S263, if it is determined that no video file is supplied to the memory 117, that is, if the data conversion portion 118 stops supplying a video file to the memory controller 116, the process proceeds to Step S268, and the control portion 119 controls the memory controller 116 to read all video files remaining in the memory 117, and allows the memory controller 116 to supply the video files to the signal process portion 115. Accordingly, the recording control is performed such that the video tree ring data having the data amount equivalent to an integral multiple of the ECC block is recorded onto as many ECC blocks as the integral multiple.

[0320]

A video file has the data amount equivalent to an integral multiple of the ECC block. Thus, at Step S268, the video tree ring data having the data amount equivalent to an integral multiple of the ECC block is recorded onto as many ECC blocks as the integral multiple.

[0321]

Thereafter, the process proceeds to Step S269. The control portion 119 sets variable Nv to a value equivalent to the infinity (a very large value), and the video file recording task then terminates.

[0322]

Accordingly, similarly to the video file recording task in FIG. 34, in the video file recording task in FIG. 36, the video tree ring data having the data amount, for example, equivalent to an integral multiple of the ECC block as units of reading and writing to the optical disk 7 is cyclically recorded onto as many ECC blocks as the integral multiple so that the boundaries of the video tree ring data match those of the ECC block on the optical disk 7.

[0323]

Next, referring to the flowchart in FIG. 39, the low resolution data recording task that starts at Step S235 in FIG. 33 to record low resolution data as a low resolution data file will be described.

[0324]

When the low resolution data recording task starts, the control portion 119 first initializes variable Nl to, for example, 1 at Step S271. Variable Nl is incremented by one in the process at Step S277 later. The process proceeds to Step S272.

[0325]

At Step S272, the control portion 119 determines whether or not Tsl x Nl is smaller than Tsa x Na, and equal or smaller than Tsv x Nv, and Tsm x Nm.

[0326]

Here, similar to the case described at Step S262 in FIG. 37, the condition that Tsl x Nl is smaller than or equal to Tsa x Na is required for recording the targeted low resolution tree ring data, that is, low resolution tree ring data to be recorded next, immediately after the audio tree ring data recorded for the most recent reproduction time slot before reproduction time Tsl x Nv. Further, similar to the case described at Step S262 in FIG. 37, the condition that Tsl x Nl is smaller than or equal to Tsv x Nv is required for recording the targeted low resolution tree ring data, that is, low resolution tree ring data to be recorded next, immediately after the video tree ring data recorded for the most recent reproduction time slot before reproduction time Tsl x Nl.

[0327]

Further, similar to the case at Step S252 in FIG. 34, the condition that Tsl x Nl is smaller than or equal to Tsm x Nm is required for recording the targeted low resolution tree ring data, that is, low resolution tree ring data to be recorded next, i.e., low resolution tree ring data for the most recent reproduction time slot before reproduction time Tsv x Nv (closest to reproduction time Tsv x Nv), immediately before the meta tree ring data for the most recent reproduction time slot before reproduction time Tsl x Nl, in other words, immediately after the meta tree ring data recorded for the next most recent reproduction time slot before reproduction time Tsl x Nl.

[0328]

At Step S272, if it is determined that reproduction time Tsl x Nl for the low resolution tree ring data is not smaller than reproduction time Tsa x Na for the audio tree ring data, not smaller than reproduction time Tsv x Nv for the video tree ring data, or not smaller than or equal to reproduction time Tsm x Nm for the meta tree ring data, that is, if the current timing is not a timing when the targeted low resolution tree ring data should be recorded, the process returns to Step S272, and the similar process is repeated thereafter.

[0329]

At Step S272, if it is determined that reproduction time Tsl x Nl for the low resolution tree ring data is smaller than reproduction time Tsa x Na for the audio tree ring data, smaller than reproduction time Tsv x Nv for the video tree ring data, and smaller than or equal to reproduction time Tsm x Nm for the meta tree ring data, that is, if the current timing is a timing when the targeted low resolution tree ring data should be recorded, the process returns to Step S273. The control portion 119 determines whether or not low resolution data is supplied to the memory 117 from the data conversion portion 118 via the memory controller 116, and when it is determined that the low resolution data is supplied, the process proceeds to Step S274.

[0330]

At Step S274, the control portion 119 determines whether or not low resolution data needed for the reproduction equivalent to low resolution tree ring size Tsl x Nl in total has been stored in the memory 117. When it is determined that sufficient low resolution data has not yet been stored in the memory 117, the process returns to Step S272, and the similar process is repeated. When it is determined at Step S274 that low resolution data equivalent to reproduction time Tsl x Nl has been stored in the memory 117, the process proceeds to Step S275.

[0331]

It should be noted that if the data amount detection portion 141 of the data conversion portion 118 notifies the memory controller 116 of the detection. Based on the notification, the memory controller 116 determines whether or not low resolution data needed for the reproduction equivalent to reproduction time Tsl x Nl in total has been stored in the memory 117, and the determination result is notified to the control portion 119. That is, the control portion 119 performs the determination at step S274 based on the determination result from the memory controller 116. It should be noted that, in this embodiment, data obtained by compressing and encoding a video file or the like whose data amount is reduced is used as the low resolution data, but a video file or the like obtained by reducing a data amount of a video file or the like may alternatively be used as it is as the low resolution data.

[0332]

At Step S275, the control portion 119 controls the memory controller 116 to extract, from low resolution data stored in the memory 117, low resolution data with the data amount that is an integral multiple of (multiplied by n) data amount Bu of one ECC block, for example, as a unit of reading or writing to the optical disk 7, and is equivalent to a maximum data amount capable of being read from the memory 117, by reading the low resolution data in the chronological order of input. The process proceeds to Step S256.

[0333]

It should be noted that the low resolution tree ring data to be read from the memory 117 as the low resolution data whose data amount is an integral multiple of the ECC block and is equivalent to a maximum data amount capable of being read from the memory 117 corresponds to the most recent low resolution tree ring data before reproduction time Tsl x Nl as mentioned above.

[0334]

Low resolution data not read at Step S275 is left in the memory 117 as it is.

[0335]

At Step S276, the control portion 119 causes the memory controller 116 to supply the signal process portion 115 with targeted low resolution tree ring data obtained at Step S275 having the data amount equivalent to an integral multiple of the ECC block. Accordingly, recording control is performed so as to record the targeted low resolution tree ring data having the data amount equivalent to an integral multiple of the ECC block onto as many ECC blocks as the integral multiple. Accordingly, the low resolution tree ring data having the data amount equivalent to an integral multiple of the ECC block is recorded onto as many ECC blocks as the integral multiple so that the boundaries of the low resolution tree ring data match those of the ECC block on the optical disk 7.

[0336]

The process proceeds to Step S277, and the control portion 119 increments variable Nl by 1. The process returns to Step S272, and the similar process is repeated.

[0337]

Meanwhile, at Step S273, if it is determined that no low resolution data is supplied to the memory 117, that is, if the data conversion portion 118 stops supplying low resolution data to the memory controller 116, the process proceeds to Step S278, and the control portion 119 controls the memory controller 116 to read all low resolution data remaining in the memory 117, and causes the memory controller 116 to supply the low resolution data to the signal process portion 115. Accordingly, recording control is performed so as to record the low resolution tree ring data having the data amount equivalent to an integral multiple of the ECC block onto as many ECC blocks as the integral multiple.

[0338]

A low resolution data file has the data amount equivalent to an integral multiple of the ECC block. Thus, at Step S278, the low resolution data tree ring data having the data amount equivalent to an integral multiple of the ECC block is recorded onto as many ECC blocks as the integral multiple.

[0339]

Thereafter, the process proceeds to Step S279, and the control portion 119 sets variable Nl to a value equivalent to the infinity (a very large value). The low resolution data recording task then terminates.

[0340]

Next, referring to the flowchart in FIG. 40, the metadata recording task initiated at Step S236 in FIG. 33 will be described.

[0341]

When the metadata recording task starts, the control portion 119 first initializes variable Nl to 1, for example, at Step S287. Variable Nl is incremented by one in the process at Step S287 later. The process then proceeds to Step S282.

[0342]

At Step S282, the control portion 119 determines whether or not Tsm x Nm is smaller than Tsa x Na, and smaller than or equal to Tsv x Nv, and Tsl x Nl.

[0343]

Here, similar to the case described at Step S262 in FIG. 37, the condition that Tsm x Nm is smaller than or equal to Tsa x Na is required for recording the targeted meta tree ring data, that is, meta tree ring data to be recorded next, immediately after the audio tree ring data recorded for the most recent reproduction time slot before reproduction time Tsm x Nm. Further, similar to the case described at Step S262 in FIG. 37, the condition that Tsm x Nm is smaller than or equal to Tsv x Nv is required for recording the targeted meta tree ring data, that is, meta tree ring data to be recorded next, immediately after the video tree ring data recorded for the most recent reproduction time slot before reproduction time Tsm x Nm.

[0344]

At Step S282, if it is determined that reproduction time Tsm x Nm for the meta tree ring data is not smaller than reproduction time Tsa x Na for the audio tree ring data, not smaller than reproduction time Tsv x Nv for the video tree ring data, or not smaller than or equal to reproduction time Tsl x Nl for the meta tree ring data, that is, if the current timing is not a timing when the targeted meta tree ring data should be recorded, the process returns to Step S282, and the similar process is repeated thereafter.

[0345]

At Step S282, if it is determined that reproduction time Tsm x Nm for the meta tree ring data is smaller than reproduction time Tsa x Na for the audio tree ring data, smaller than reproduction time Tsv x Nv for the video tree ring data, and smaller than or equal to reproduction time Tsl x Nl for the low resolution tree ring data, that is, if the current timing is a timing when the targeted meta tree ring data should be recorded, the process returns to Step S283. The control portion 119 determines whether or not metadata is supplied to the memory 117 from the data conversion portion 118 via the memory controller 116, and when it is determined that the metadata is supplied, the process proceeds to Step S284.

[0346]

At Step S284, the control portion 119 determines whether or not metadata needed for the reproduction equivalent to meta tree ring size Tsm x Nm in total has been stored in the memory 117. When it is determined that sufficient metadata has not yet been stored in the memory 117, the process returns to Step S282, and the similar process is repeated. Further, when it is determined at Step S284 that the metadata equivalent to reproduction time Tsm x Nm has been stored in the memory 117, the process proceeds to Step S285.

[0347]

If video files and audio files needed for the reproduction equivalent to reproduction time Tsm x Nm in total are detected, the data amount detection portion 141 of the data conversion portion 118 notifies the memory controller 116 of the detection. Based on the notification, the memory controller 116 determines whether or not metadata needed for the reproduction equivalent to reproduction time Tsm x Nm in total is stored in the memory 117, and the determination result is notified to the control portion 119. Then, the control portion 119 performs the determination at step S284 based on the determination result from the memory controller 116.

[0348]

At Step S285, the control portion 119 controls the memory controller 116 to extract, from metadata stored in the memory 117, metadata with the data amount that is an integral multiple of (multiplied by n) data amount Bu of one ECC block, for example, as a unit of reading or writing to the optical disk 7, and is equivalent to a maximum data amount capable of being read from the memory 117, by reading the metadata in the chronological order of input. The process proceeds to Step S286.

[0349]

It should be noted that the meta tree ring data to be read from the memory 117 as the metadata whose data amount is an integral multiple of the ECC block and is equivalent to a maximum data amount capable of being read from the memory 117 corresponds to the most recent meta tree ring data before reproduction time Tsm x Nm as mentioned above.

[0350]

Metadata not read at Step S285 is left in the memory 117 as it is.

[0351]

At Step S286, the control portion 119 allows the memory controller 116 to supply the signal process portion 115 with targeted meta tree ring data obtained at Step S285 having the data amount equivalent to an integral multiple of the ECC block. Accordingly, recording control is performed so as to record the targeted meta tree ring data having the data amount equivalent to an integral multiple of the ECC block onto as many ECC blocks as the integral multiple. Accordingly, the meta tree ring data having the data amount equivalent to an integral multiple of the ECC block is recorded onto as many ECC blocks as the integral multiple so that the boundaries of the meta tree ring data match those of the ECC block on the optical disk 7.

[0352]

Thereafter, the process proceeds to Step S289, and the control portion 119 increments variable Nm by 1. The process returns to Step S282, and the similar process is repeated.

[0353]

Meanwhile, at Step S283, if it is determined that no metadata is supplied to the memory 117, that is, if the data conversion portion 118 stops supplying metadata to the memory controller 116, the process proceeds to Step S288. The control portion 119 controls the memory controller 116 to read all metadata remaining in the memory 117, and causes the memory controller 116 to supply the metadata to the signal process portion 115. Accordingly, recording control is performed so as to record the meta tree ring data having the data amount equivalent to an integral multiple of the ECC block onto as many ECC blocks as the integral multiple.

[0354]

Thereafter, the process proceeds to Step S289. The control portion 119 sets variable Nm to a value equivalent to the infinity, and the metadata recording task then terminates.

[0355]

In this manner, the process for the audio file recording task, the video file recording task, the low resolution data recording task, and the metadata recording task is performed to record audio files, video files, metadata, and low resolution data on the optical disk 7. Accordingly, for example, if audio tree ring size Tsa and video tree ring size Tsv are equivalent to the same time, the audio tree ring data as a set of audio files and the video tree ring data as a set of video files for similar reproduction time slots are sequentially recorded so that they are arranged at adjacent positions on the optical disk 7. Further, the low resolution tree ring data as a set of low resolution data and the metadata tree ring data as a set of metadata for similar reproduction time slots are sequentially recorded so that they are arranged at positions adjacent to the audio tree ring data and the video tree ring data on the optical disk 7.

[0356]

Video files, audio files, and the like are recorded on the optical disk 7 as if tree rings were formed. From this analogy, a set of audio files or video files recorded on the optical disk 7 is referred to as audio "tree ring" data or video "tree ring" data. The same applies to the low resolution tree ring data and the meta tree ring data. Hereinafter, the term "tree ring data" is used to express a collection of data in a given data series recorded on the optical disk 7 as if tree rings were formed.

[0357]

Here, the width of a tree ring (the number of tracks to record one piece of audio tree ring data or video tree ring data) formed on the optical disk 7 is determined by audio tree ring size Tsa and video tree ring size Tsv. It should be noted that audio tree ring size Tsa and video tree ring size Tsv can be varied in accordance with radial positions in the optical disk 7 on which the audio tree ring data and the video tree ring data. Depending on audio tree ring size Tsa or video tree ring size Tsv, there may be a case of recording one piece of audio tree ring data or video tree ring data on a track that is shorter than one lap.

[0358]

As mentioned above, audio tree ring data and video tree ring data for similar reproduction time slots are recorded at close positions on the optical disk 7. Therefore, it is possible to fast read and reproduce an audio file and a video file at the same reproduction time from the optical disk 7.

[0359]

Further, since the audio file and the video file are assumed to be tree ring data equivalent to the data amount of multiple ECC blocks, and the audio file and the video file are recorded on the multiple ECC blocks so that boundaries of the tree ring data match those of ECC blocks, it is possible to read only the audio file or the video file from the optical disk 7, with a result that it is possible to fast edit only the audio file or the video file.

[0360]

Because the header, the body, and the footer in a video file each are assigned with the data amount equal to an integral multiple of an ECC block, the header, the body, and the footer are recorded in units of ECC blocks. That is, one ECC block does not record any two of the header, the body, and the footer.

[0361]

Therefore, when one of the header, the body, and the footer is written or read, the write or read process is performed for the minimum number of ECC blocks, thereby enabling a more efficient write or read process. As a result, in a file writing process, the number of clusters whose data is to be rewritten becomes minimum. When the optical disk 7 is subject to a physical limitation (in terms of physicality) on the number of rewrite operations, there is an advantage of prolonging the life of the optical disk 7 with respect to the number of rewrite operations.

[0362]

The determination processes that are performed at Step S252 of the audio file recording task in FIG. 34, Step S262 of the video file recording task in FIG. 37, Step S272 of the low resolution data recording task in FIG. 39, and Step S282 of the metadata recording task in FIG. 40 allow the audio tree ring data, the video tree ring data, the low resolution tree ring data, and the meta tree ring data for similar reproduction time slots to be collectively and cyclically recorded on the optical disk 7 in the order of the audio tree ring data, the video tree ring data, the low resolution tree ring data, and the meta tree ring data.

[0363]

It should be noted that the priority of recording on the optical disk 7 is not limited to the order of the audio tree ring data, the video tree ring data, the low resolution tree ring data, and the meta tree ring data as mentioned above.

[0364]

For example, the priority of recording on the optical disk 7 may be the order of the meta tree ring data, the audio tree ring data, the video tree ring data, and the low resolution tree ring data.

[0365]

Next, as mentioned above, the memory controller 116 reads data from the memory 117 to extract the audio tree ring data, the video tree ring data, the low resolution tree ring data, and the meta tree ring data. The following further describes processes to configure (extract) the audio tree ring data, the video tree ring data, the low resolution tree ring data, and the meta tree ring data with reference to FIGS. 41 through 45.

[0366]

FIG. 41 shows the relationship between time (reproduction time) t and each of total data amount (total data amount) La for audio files, total data amount Lv for video files, total data amount Ll for low resolution data, and total data amount Lm for metadata. On the right of FIG. 41 (also on FIGS. 42 through 45 described later), a small vertical bidirectional arrow (an arrow indicating an interval between horizontal dotted lines) represents data amount Bu of an ECC block.

[0367]

As mentioned above, when audio files needed for the reproduction equivalent to reproduction time Tsa x Na are stored in the memory 117, the memory controller 116 reads an audio file having the maximum data amount that is capable of reading from the memory 117 and is equivalent to an integral multiple of the ECC block, and extracts this audio file as audio tree ring data. When video files needed for the reproduction equivalent to reproduction time Tsv x Nv are stored in the memory 117, the memory controller 116 reads a video file having the maximum data amount that is capable of reading from the memory 117 and is equivalent to an integral multiple of the ECC block, and extracts this video file as video tree ring data. When low resolution data needed for the reproduction equivalent to reproduction time Tsl x Nl is stored in the memory 117, the memory controller 116 reads low resolution data having the maximum data amount that is capable of reading from the memory 117 and is equivalent to an integral multiple of the ECC block, and extracts this low resolution data as low resolution tree ring data. Further, when metadata needed for the reproduction equivalent to reproduction time Tsm x Nm is stored in the memory 117, the memory controller 116 reads metadata having the maximum data amount that is capable of reading from the memory 117 and is equivalent to an integral multiple of the ECC block, and extracts this metadata as meta tree ring data.

[0368]

Therefore, as shown in FIG. 41, if total data amount La of an audio file stored in the memory 117 changes, as shown in FIG. 42, the memory controller 116 reads the audio file having the maximum data amount capable of reading from the memory 117 and equivalent to an integral multiple of the ECC block at the timing when the time T becomes i x Tsa (where i = 1, 2, and so on) equivalent to an integral multiple of video tree ring size, and extracts this audio file as audio tree ring data.

[0369]

In the embodiment of FIG. 42, the audio file is extracted for one ECC block, two ECC blocks, one ECC block, and two ECC blocks as audio tree ring data #1, #2, #3, and #4 at timings when the time t becomes Tsa, 2 x Tsa, 3 x Tsa, and 4 x Tsa, respectively.

[0370]

Further, as shown in FIG. 41, if total data amount Lv of a video file stored in the memory 117 changes, as shown in FIG. 43, the memory controller 116 reads the video file having the maximum data amount capable of reading from the memory 117 and equivalent to an integral multiple of the ECC block at the timing when the time t becomes i x Tsv equivalent to an integral multiple of video tree ring size Tsv, and extracts the read video file as video tree ring data.

[0371]

According to the embodiment in FIG. 43, the video file is extracted for four ECC blocks, two ECC blocks, five ECC blocks, and two ECC blocks as video tree ring data #1, #2, #3, and #4 at timings when the time t becomes Tsv, 2 x Tsv, 3 x Tsv, and 4 x Tsv, respectively.

[0372]

Further, as shown in FIG. 41, if total data amount Ll of low resolution data stored in the memory 117 changes, as shown in FIG. 44, the memory controller 116 reads the low resolution data having the maximum data amount capable of reading from the memory 117 and equivalent to an integral multiple of the ECC block at the timing when the time t becomes i x Tsl equivalent to an integral multiple of low resolution tree ring size Tsl, and extracts the read low resolution data as low resolution tree ring data.

[0373]

In the embodiment of FIG. 44, the low resolution data is extracted for one ECC block and three ECC blocks as low resolution tree ring data #1 and #2 at timings when the time t becomes Tsl and 2 x Tsl, respectively.

[0374]

Further, as shown in FIG. 41, if total data amount Lm of metadata stored in the memory 117 changes, as shown in FIG. 45, the memory controller 116 reads the metadata having the maximum data amount capable of reading from the memory 117 and equivalent to an integral multiple of the ECC block at the timing when the time t becomes i x Tsm equivalent to an integral multiple of meta tree ring size Tsm, and extracts the read metadata as meta tree ring data.

[0375]

In the embodiment of FIG. 45, the metadata is extracted for each one ECC block as meta tree ring data #1 and #2 at timings when the time t becomes Tsm and 2 x Tsm, respectively.

[0376]

For example, let us suppose such relationship between audio tree ring size Tsa in FIG. 42, video tree ring size Tsv in FIG. 43, low resolution tree ring size Tsl in FIG. 44, and meta tree ring size Tsm in FIG. 45 that video tree ring size Tsv equals audio tree ring size Tsa and low resolution tree ring size Tsl and meta tree ring size Tsm equals the double of audio tree ring size Tsa (2 x Tsa=2 x Tsv=Tsl=Tsm), for example. In this case, the audio file recording task in FIG. 34, the video file recording task in FIG. 37, the low resolution data recording task in FIG. 39, and the metadata recording task in FIG. 40 cyclically record the audio tree ring data #1 through #4 in FIG. 42, the video tree ring data #1 through #4 in FIG. 43, the low resolution tree ring data #1 and #2 in FIG. 44, and the meta tree ring data #1 and #2 in FIG. 45 on the optical disk 7 as shown in FIG. 27.

[0377]

That is, the audio tree ring data, the video tree ring data, the low resolution tree ring data, and the meta tree ring data for similar reproduction time slots are recorded at positions toward the beginning of the optical disk 7 according to the priority of the audio tree ring data, the video tree ring data, the low resolution tree ring data, and the meta tree ring data as mentioned above.

[0378]

With respect to the audio tree ring data having the highest priority, for example, the video tree ring data with the same video tree ring size Tsv as audio tree ring size Tsa is recorded on the optical disk 7 at the same cycle as the audio tree ring data. That is, when audio tree ring data is recorded for a given reproduction time slot, the audio tree ring data is followed by video tree ring data recorded for a reproduction time slot similar to that of the recorded audio tree ring data.

[0379]

In addition, a low resolution tree ring having low resolution tree ring size Tsl that is twice as large as audio tree ring size Tsa is recorded on the optical disk 7 at a cycle that is twice as large as that of audio tree ring data. That is, low resolution tree ring data for a given reproduction time slot corresponds to audio tree ring data having two reproduction time slots that halve the given reproduction time slot. The low resolution tree ring data is recorded after the audio tree ring data for the two reproduction time slots is recorded.

[0380]

Further, a meta tree ring having meta tree ring size Tsm that is twice as large as audio tree ring size Tsa is recorded on the optical disk 7 at a cycle that is twice as large as that of audio tree ring data. That is, meta tree ring data for a given reproduction time slot corresponds to audio tree ring data having two reproduction time slots that halve the given reproduction time slot. The meta tree ring data is recorded after the audio tree ring data for the two reproduction time slots is recorded.

[0381]

Consequently, as shown in FIG. 46, the audio tree ring data #1 through #4 in FIG. 42, the video tree ring data #1 through #4 in FIG. 43, the low resolution tree ring data #1 and #2 in FIG. 44, and the meta tree ring data #1 and #2 in FIG.

45 are recorded on the optical disk 7 from inside peripheries to outside peripheries thereof in the order of audio tree ring data #1, video tree ring data #1, audio tree ring data #2, video tree ring data #2, low resolution tree ring data #1, meta tree ring data #1, audio tree ring data #3, video tree ring data #3, audio tree ring data #4, video tree ring data #4, low resolution tree ring data #2, meta tree ring data #2, and so on.

[0382]

In the embodiment as shown in FIGS. 41 through 46, video tree ring size Tsv is assumed to be equal to audio tree ring size Tsa, and low resolution tree ring size Tsl or meta tree ring size Tsm is assumed to be equal to the double of audio tree ring size Tsa, but the relationship between audio tree ring size Tsa, video tree ring size Tsv, low resolution tree ring size Tsl, and meta tree ring size Tsm is not limited thereto. That is, audio tree ring size Tsa, video tree ring size Tsv, low resolution tree ring size Tsl, and meta tree ring size Tsm all may be the same time or different times.

[0383]

Further, audio tree ring size Tsa, video tree ring size Tsv, low resolution tree ring size Tsl, and meta tree ring size Tsm can be set in accordance with uses and purposes of the optical disk 7.

[0384]

That is, low resolution tree ring size Tsl and meta tree ring size Tsm can be larger than audio tree ring size Tsa and video tree ring size Tsv.

[0385]

Let us consider that low resolution tree ring size Tsl is larger than audio tree ring size Tsa and video tree ring size Tsv (e.g., low resolution tree ring size Tsl is ten seconds while audio tree ring size Tsa and video tree ring size Tsv each are two seconds). In such case, for example, it is possible to improve shuttle reproduction rates using low resolution data and transfer rates of low resolution data to external apparatuses such as computers.

[0386]

In other words, the low resolution data has smaller data amount than mainline data and therefore is capable of fast reading from the optical disk 7. Further, the low resolution data causes small processing loads and thus can be used for variable speed reproduction such as the shuttle reproduction. Further, when increasing low resolution tree ring size Tsl, the frequency of seeks that occur during reading of only low resolution data from the optical disk 7 can be reduced, with the result that it becomes possible to faster read only the low resolution data from the optical disk 7 and improve speeds of shuttle reproduction when performing the shuttle reproduction using the low resolution data. Further, when the low resolution data is transferred to a computer for processing, a transfer rate can be improved (to shorten the time required for the transfer).

[0387]

Further, when meta tree ring size Ts is larger than audio tree ring size Tsa and video tree ring size Tsv (e.g., meta tree ring size Tsm is 20 seconds while audio tree ring size Tsa and video tree ring size Tsv each are two seconds), similarly to the case of increasing low resolution tree ring size Tsl, only metadata can be read from the optical disk 7 in a short period of time. Accordingly, for example, a time code included in the metadata can be used to retrieve a specific frame from a video file as mainline data at high speed.

[0388]

Therefore, when there is a need for the shuttle reproduction or fast transfer of low resolution data to the outside, low resolution tree ring size Tsl is increased, and when there is a need for fast frame retrieval, meta tree ring size Ts is increased, with the result that it becomes possible to provide the highly convenient optical disk 7 that satisfies these needs.

[0389]

As mentioned above, by increasing low resolution tree ring size Tsl or meta tree ring size Tsm, the time required to read (or even to write) a specific data series of only low resolution data, metadata or the like can be shortened.

[0390]

Accordingly, when increasing audio tree ring size Tsa or video tree ring size Tsv, the time required to read (or even to write) only audio files or video files as mainline data can be shortened. As a result, when performing so-called AV (Audio Visual) split editing that edits only audio files or video files, it is possible to accelerate the process thereof.

[0391]

However, when the video and the audio are reproduced, a wait is needed until there become available video files corresponding to the reproduction times and audio files associated with the video files. When audio tree ring size Tsa or video tree ring size Tsv is increased, it is necessary to read an audio file having the increased audio tree ring size Tsa or a video file having the increased video tree ring size Tsv and then read the other. As a result, the time lapse increases until the video file at a given reproduction time and the audio file associated with the video file become available, and this also increases a delay between the time the reproduction is instructed and the time the reproduction actually starts. Further, in a case of simultaneously reproducing the video file at a given reproduction time and the audio file associated with the video file, the audio file having the increased audio tree ring size Tsa or the video file having the increased video tree ring size Tsv is read first and needs to be stored in the memory 117 at least until the other starts being read. In consideration for the foregoing, increasing audio tree ring size Tsa or video tree ring size Tsv also increases a delay until the start of reproduction and necessitates a large capacity for the memory 117.

[0392]

Accordingly, it is desirable to determine audio tree ring size Tsa and video tree ring size Tsv in consideration for a delay time until the start of reproduction and allowable values for the capacity of the memory 117.

[0393]

It should be noted that increasing low resolution tree ring size Tsl or meta tree ring size Tsm causes a permissible increase in the capacity needed for the memory 117 compared to the case of increasing audio tree ring size Tsa or video tree ring size Tsv, because the low resolution data or the metadata has a data amount sufficiently smaller than that of the audio file or the video file..

[0394]

In addition, the priority of recording on the optical disk 7 may be meta tree ring data, audio tree ring data, video tree ring data, and low resolution tree ring data in order. In this case, as shown in FIG. 47, for example, meta tree ring data #1 and #2, audio tree ring data #1 and #2, video tree ring data #1 and #4, and low resolution tree ring data #1 and #2 are recorded on the optical disk 7 from inside peripheries to outside peripheries thereof in the order of meta tree ring data #1, audio tree ring data #1, video tree ring data #1, low resolution tree ring data #1, meta tree ring data #2, audio tree ring data #2, video tree ring data #2, low resolution tree ring data #2, and so on.

[0395]

FIG. 48 shows how the disk drive apparatus 11 reads or writes data on the optical disk 7. In FIG. 48, it is assumed that reading or writing of four data series of a metadata, an audio file, a video file, and low resolution data on the optical disk 7 is performed.

[0396]

In FIG. 48, meta tree ring data #1, audio tree ring data #1, video tree ring data #1, and low resolution tree ring data #1 are represented as tree ring data #1; meta tree ring data #2, audio tree ring data #2, video tree ring data #2, and low resolution tree ring data #2 are represented as tree ring data #2; and similarly, Nth data, i.e., meta tree ring data #N, audio tree ring data #N, video tree ring data #N, and low resolution tree ring data #N are represented as tree ring data #N.

[0397]

When data is written to the optical disk 7,if the optical disk has a sufficient contiguous free area having no flaw (defect), audio tree ring data, video tree ring data, low resolution tree ring data, and meta tree ring data extracted from data series of the metadata, the audio file, the video file, and the low resolution data, respectively, are written to the free area on the optical disk 7 like a single stroke as shown in FIG. 48. It should be noted that the meta tree ring data, the audio tree ring data, the video tree ring data, and the low resolution tree ring data each have the data amount equivalent to an integral multiple of the ECC block of the optical disk 7, and are recorded so that the data boundaries thereof match the ECC block boundaries.

[0398]

As described with reference to the flowchart for the metadata file generation process in FIG. 13 and the flowchart for the video file generation process in FIG. 15, the metadata file and the video file are supplied in the order of the body, the footer, and the header to the disk drive apparatus 11.

[0399]

As described with reference to the flowchart for the audio file generation process in FIG. 16, the audio file is supplied to the disk drive apparatus 11 in the order of the value and the KLV-structured filler as the body's audio items, then the footer, and then the header, the audio item key, and the length.

[0400]

As described with reference to the flowchart for the low resolution file synthesis process in FIG. 32, the low resolution file is supplied to the memory controller 116 in the order of the body, the footer, and the header.

[0401]

Consequently, audio tree ring data, video tree ring data, low resolution tree ring data, and meta tree ring data extracted from the data series of the metadata, the audio file, the video file, and the low resolution data are written to a free area on the optical disk 7 in the order of the body, the footer, and the header as shown in FIG. 48.

[0402]

The processes described with reference to the flowcharts for the metadata file generation process in FIG. 13, the video file generation process in FIG. 15, the audio file generation process in FIG. 16, the low resolution file synthesis process in FIG. 32, and the recording process in FIG. 33 will be collectively described as a recording process with reference to a flowchart in Fig. 49.

[0403]

At Step S291, the control portion 119 of the disk drive apparatus 11 records bodies of the metadata file, the video file, the audio file, and the low resolution file on the optical disk 7, and the process proceeds to Step S292. At Step S292, the control portion 119 determines whether or not the bodies of the metadata file, the video file, the audio file, and the low resolution file have been recorded completely. When it is determined that the body recording is not completed, the process returns to Step S291. The body recording process is repeated.

[0404]

When it is determined at Step S292 that the body recording is completed, the process proceeds to Step S293. The control portion 119 records footers of the metadata file, the video file, the audio file, and the low resolution file on the optical disk 7, and the process proceeds to Step S294. At Step S294, the control portion 119 determines whether or not the footers of the metadata file, the video file, the audio file, and the low resolution file have been recorded completely. When it is determined that the footer recording is not completed, the process returns to Step S293, and the footer recording process is repeated.

[0405]

When it is determined at Step S294 that the footer recording is completed, the process proceeds to Step S295. The control portion 119 records headers of the metadata file, the video file, the audio file, and the low resolution file on the optical disk 7, then the recording process then terminates.

[0406]

As mentioned above, since the header is recorded after the body and the footer, a single process can be used to record the header containing data such as the audio data reproduction time or the time code (TC) that is determined by settlement of the body.

[0407]

Moreover, it is possible to reliably record the header following the body and the footer, i.e., at positions close to the body and the footer on the optical disk 7.

[0408]

When a file is read from the optical disk 7, the header, the body, and the footer are sequentially sought and read.

[0409]

Further, in this embodiment, in the memory controller 116, the audio tree ring data is extracted by reading an audio file at every time corresponding to an integral multiple of audio tree ring size Tsa so that the audio file's data amount is an integral multiple of the read or write unit such as an ECC block and is equivalent to a maximum data amount capable of being read from the memory 117. That is, when an audio file with the data amount that is greater than N ECC blocks and is smaller than N+1 ECC blocks is stored in the memory 117 at the time corresponding to an integral multiple of audio tree ring size Tsa, an audio file with the data amount of N ECC blocks is extracted as audio tree ring data. In addition, for example, after the time reaches an integral multiple of audio tree ring size Tsa, it is possible to extract audio tree ring data by waiting until an audio file with the data amount greater than or equal to N+1 ECC blocks is stored in the memory 117, and then reading the audio file with the data amount equivalent to N+1 ECC blocks. The same applies to extraction of the video tree ring data, the low resolution tree ring data, and the meta tree ring data. That is, the data amount of tree ring data just needs to be equivalent to an integral multiple of the unit of reading or writing to the optical disk 7 and approximate to the data amount needed for the reproduction equivalent to the reproduction time specified as the audio tree ring size and the like.

[0410]

All the constituent elements of metadata can be included in the meta tree ring data. Further, some of the constituent elements can be included in the meta tree ring data and the other constituent elements can be recorded independently of the meta tree ring data. That is, the metadata constituent elements are grouped into, for example, constituent elements such as the time code usable for the retrieval of video file frames and the other constituent elements. The constituent elements usable for the retrieval can be collectively recorded on inside peripheries of the optical disk 7, for example. The other constituent elements can be included in the meta tree ring data and cyclically recorded on the optical disk 7. In this case, since the constituent elements usable for the retrieval are collectively recorded on the optical disk 7, the time needed for the retrieval can be shortened.

[0411]

All constituent elements of the metadata may be collectively recorded on inside peripheries of the optical disk 7. When all constituent elements of the metadata are collectively recorded on inside peripheries of the optical disk 7, for example, it is necessary to suspend the recording of data series other than the metadata until all constituent elements of the metadata are completely recorded. Alternatively, all constituent elements of the metadata need to be stored until data series other than the metadata are completely recorded. By contrast, when only metadata's constituent elements usable for the retrieval are collectively recorded, it is possible to shorten the time to wait until recording of the data series other than the metadata compared to the case of collectively recording all the constituent elements of the metadata on the optical disk 7. Alternatively, it is possible to reduce the data amount of metadata that needs to be stored until the data series other than the metadata are completely recorded.

[0412]

The present invention is applicable to disk-shaped recording media other than optical disks.

[0413]

While there have been described the cases of arranging video files and audio files on spiral tracks, the files can be alternately arranged on concentrical tracks. In this case, tracks continue from an inner one to an outer one.

[0414]

Next, FIG. 50 shows an exemplary configuration of the independent/standard format conversion portion 22 in FIG. 7.

[0415]

A buffer 301 temporarily stores AV independent format files (master file, file-based metadata file, frame-based metadata file, auxiliary file, video file, audio files for eight channels, and low resolution data file) supplied from the disk drive apparatus 11 (FIG. 1).

[0416]

A file acquisition portion 302 references the master file stored in the buffer 301 to identify names of the file-based metadata file, the frame-based metadata file, the auxiliary file, the video file, the audio files for eight channels, and the low resolution file. Based on the file names, the file acquisition portion 302 obtains the file-based metadata file, the frame-based metadata file, the auxiliary file, the video file, the audio files for eight channels, and the low resolution file via the buffer 301 by causing the disk drive apparatus 11 to read these files. Further, the file acquisition portion 302 supplies the obtained files to the corresponding file process portions, i.e., the file-based metadata file and the frame-based metadata file to a metadata file process portion 303; the auxiliary file to an auxiliary file process portion 304; the video file to a video file process portion 305; and the audio files for eight channels to an audio file process portion 306. The file acquisition portion 302 supplies the low resolution file to a buffer 309.

[0417]

The metadata file process portion 303 extracts file-based metadata from the file-based metadata file supplied from the file acquisition portion 302, extracts a system item in which frame-based metadata is placed from the frame-based metadata file, and supplies the file-based metadata and the system item to the data synthesis portion 307.

[0418]

The auxiliary file process portion 304 extracts an auxiliary item from the auxiliary file supplied from the file acquisition portion 302 and supplies the auxiliary item to the data synthesis portion 307.

[0419]

The video file process portion 305 extracts a picture item from the video file supplied from the file acquisition portion 302 and supplies the picture item to the data synthesis portion 307.

[0420]

The audio file process portion 105 extracts channel-based audio data from the audio files for eight channels supplied from the file acquisition portion 302. Further, the audio file process portion 105 multiplexes and arranges the channel-based audio data to configure a sound item and supplies it to the data synthesis portion 307.

[0421]

The data synthesis portion 307 configures a standard AV multiplexing format file using the file-based metadata and the system item supplied from the metadata file process portion 303, the auxiliary item supplied from the auxiliary file process portion 304, the picture item supplied from the video file process portion 305, and the sound item supplied from the audio file process portion 306 and supplies the standard AV multiplexing format file to a buffer 308.

[0422]

The buffer 308 temporarily stores the standard AV multiplexing format file supplied from the data synthesis portion 307 or the low resolution file supplied from the file acquisition portion 302 and supplies the file to the communication I/F 13 (FIG. 1).

[0423]

FIG. 51 shows an exemplary configuration of the video file process portion 305 in FIG. 50.

[0424]

The video file supplied from the file acquisition portion 302 is supplied to a header/footer removal portion 311. The header/footer removal portion 311 removes the header and the footer from the supplied video file and supplies the remaining body to a decomposition portion 312. The decomposition portion 312 separates a picture item sequence placed in the body supplied from the header/footer removal portion 311, thereby extracting units of multiplexing with the other items (system item, sound item, and auxiliary item), i.e., individual picture items in which frame-based video data is placed. The decomposition portion 312 supplies the picture items to the data synthesis portion 307 (FIG. 50).

[0425]

FIG. 52 shows an exemplary configuration of the audio file process portion 306 in FIG. 50.

[0426]

The audio files for eight channels supplied from the file acquisition portion 302 are supplied to a header/footer removal portion 321. The header/footer removal portion 321 removes the header and the footer from each of the supplied audio files for eight channels and supplies the resultant remaining body for each channel to a KLV decoder 322.

[0427]

The KLV decoder 322 decomposes the KLV structure of each channel's body supplied from the header/footer removal portion 321 to obtain WAVE format audio data for each channel, and supplies the WAVE format audio data to a data conversion portion 323.

[0428]

The data conversion portion 323 applies a conversion process which is reverse to that of the data conversion portion 63 in FIG. 10 to the audio data supplied from the KLV decoder 322. . That is, the data conversion portion 323 converts channel-based WAVE format audio data supplied from the KLV decoder 322 into channel-based AES3 format audio data and supplies the audio data to a channel multiplexing portion 324.

[0429]

The channel multiplexing portion 324 multiplexes channel-based audio data supplied from the data conversion portion 124 in units of samples. The channel multiplexing portion 324 supplies the resulting multiplexed audio data to a KLV encoder 325.

[0430]

The KLV encoder 325 divides the multiplexed audio data supplied from the channel multiplexing portion 324 into units corresponding to frames of the video data, and KLV-codes the multiplexed audio data corresponding to each frame according to the KLV structure. Further, the KLV encoder 325 configures a sound item by adding a filler KLV structure for complementing an insufficient fixed length of the sound item to the KLV structure of the multiplexed audio data corresponding to each frame, and supplies it to the data synthesis portion 307 (FIG. 50).

[0431]

FIG. 53 shows an exemplary configuration of the data synthesis portion 307 in FIG. 50.

[0432]

A header/footer generation portion 331 is supplied with file-based metadata output from the metadata file process portion 303. The header/footer generation portion 331 generates a header and a footer for a standard AV multiplexing format file. Further, the header/footer generation portion 331 places the file-based metadata from the metadata file process portion 303 in the header's header metadata and supplies the header and the footer to a header/footer adding portion 333.

[0433]

A multiplexing portion 332 is supplied with the system item output from the metadata file process portion 303, the auxiliary item output from the auxiliary file process portion 304, the picture item output from the video file process portion 305, and the sound item output from the audio file process portion 306. The multiplexing portion 332 sequentially multiplexes the system item, the picture item, the sound item, and the auxiliary item in this order as supplied to configure an edit unit sequence, and supplies the edit unit sequence as a body to the header/footer adding portion 333.

[0434]

The header/footer adding portion 333 adds the header and the footer supplied from header/footer generation portion 331 to the body supplied from the multiplexing portion 332. Accordingly, the header/footer a standard AV multiplexing format file is configured to be output.

[0435]

The independent/standard format conversion portion 22 in FIG. 50

performs the metadata file process for processing metadata files, the auxiliary file process for processing auxiliary files, the video file process for processing video files, and the audio file process for processing audio files, and uses the results of these processes to perform a synthesis process that synthesizes'(generates) a standard AV multiplexing format file.

[0436]

Referring now to flowcharts in FIGS. 54 through 58, the following describes the metadata file process, the auxiliary file process, the video file process, the audio file process, and the synthesis process performed by the independent/standard format conversion portion 22.

[0437]

First, with reference to the flowchart in FIG. 54, the metadata file process will be described.

[0438]

The metadata file process starts, for example, when the disk drive apparatus 11 reads the master file from the optical disk 7 and stores the master file in the buffer 301.

[0439]

That is, at Step S301, the file acquisition portion 302 references the master file stored in the buffer 301 to identify the file names of file-based and frame-based metadata files. Further, at Step S301, the file acquisition portion 302 obtains the file-based and frame-based metadata files based on the file names via the buffer 301 by causing the disk drive apparatus 11 to read those files from the optical disk 7, and supplies the file-based and frame-based metadata files to the metadata file process portion 303. The process proceeds to Step S302. At Step S302, the metadata file process portion 303 extracts file-based metadata from the file-based metadata file supplied from the file acquisition portion 302, extracts the system item containing the frame-based metadata from the frame-based metadata file, and supplies the file-based metadata and the system item to the data synthesis portion 307 to terminate the metadata file process.

[0440]

Next, with reference to the flowchart in FIG. 55, the auxiliary file process will be described.

[0441]

The auxiliary file process starts, for example, when the disk drive apparatus 11 reads the master file from the optical disk 7 and stores the master file in the buffer 301.

[0442]

That is, at Step S311, the file acquisition portion 302 first references the master file stored in the buffer 301 to identify the file name of an auxiliary file. Further, at Step S311, the file acquisition portion 302 obtains the auxiliary file based on the file name via the buffer 301 by causing the disk drive apparatus 11 to read that file from the optical disk 7, and supplies the auxiliary file to the auxiliary file process portion 304. The process proceeds to Step S312.

[0443]

At Step S312, the auxiliary file process portion 304 decomposes the auxiliary file supplied from the file acquisition portion 302 in units of auxiliary items to extract (obtain) auxiliary items from the auxiliary file, and supplies the auxiliary items to the data synthesis portion 307 to terminate the auxiliary file process.

[0444]

Next, with reference to the flowchart in FIG. 56, the video file process will be described.

[0445]

The video file process starts, for example, when the disk drive apparatus 11 reads the master file from the optical disk 7 and stores the master file in the buffer 301.

[0446]

That is, at Step S321, the file acquisition portion 302 first references the master file stored in the buffer 301 to identify the file name of a video file. Further, at Step S321, the file acquisition portion 302 obtains the video file based on the file name via the buffer 301 by causing the disk drive apparatus 11 to read that file from the optical disk 7, and supplies the video file to the video file process portion 305. The process proceeds to Step S322.

[0447]

At Step S322, the header/footer removal portion 311 of the video file process portion 305 (FIG. 51) removes the header and the footer from the video file supplied from the file acquisition portion 302, and supplies the remaining body to the decomposition portion 312. The process proceeds to Step S323. At Step S323, the decomposition portion 312 decomposes a sequence of picture items arranged in the body supplied from header/footer removal portion 311 into individual picture items and supplies these picture items to the data synthesis portion 307 to terminate the video file process.

[0448]

Next, with reference to the flowchart in FIG. 57, the audio file process will be described.

[0449]

The audio file process starts, for example, when the disk drive apparatus 11 reads the master file from the optical disk 7 and stores the master file in the buffer 301.

[0450]

That is, at Step S331, the file acquisition portion 302 first references the master file stored in the buffer 301 to identify the file name of an audio file for each of eight channels. Further, at Step S331, the file acquisition portion 302 obtains the audio files for eight channels based on the file names via the buffer 301 by causing the disk drive apparatus 11 to read those files from the optical disk 7. The file acquisition portion 302 supplies the audio files to the audio file process portion 306. The process proceeds to Step S332.

[0451]

At Step S332, the header/footer removal portion 321 of the audio file process portion 106 (FIG. 52) removes the headers and the footers from the audio files for eight channels supplied from the file acquisition portion 302, supplies the resultant remaining body for each channel to the KLV decoder 322. The process proceeds to Step S333. At Step S333, the KLV decoder 322 decomposes the KLV structure of each channel's body supplied from the header/footer removal portion 321. The KLV decoder 322 supplies the resulting WAVE format audio data for each channel to the data conversion portion 323. The process proceeds to Step S334.

[0452]

At Step S334, the data conversion portion 323 converts each channel's WAVE format audio data supplied from KLV decoder 322 into channel-based AES3 format audio data and supplies this data to the channel multiplexing portion 324. The process proceeds to Step S335. At Step S335, the channel multiplexing portion 324 multiplexes each channel's audio data supplied from the data conversion portion 124 and supplies the resulting multiplexed audio data to the KLV encoder 325. The process proceeds to Step S336.

[0453]

At Step S336, the KLV encoder 325 divides the multiplexed audio data supplied from the channel multiplexing portion 324 into units corresponding to the frames of the video data. The KLV encoder 325 KLV-codes the multiplexed audio data corresponding to the frame according to the KLV structure. The process proceeds to Step S337. Further, at Step S337, the KLV encoder 325 adds the necessary filler's KLV structure to the KLV structure of the multiplexed audio data corresponding to each frame. Accordingly, the KLV encoder 325 configures a sound item and supplies it to the data synthesis portion 307 to terminate the audio file process.

[0454]

Next, with reference to the flowchart in FIG. 58, the synthesis process will be described.

[0455]

The synthesis process starts, for example, when the data synthesis portion 307 is supplied with the file-based metadata and the system item from the metadata file process portion 303, the auxiliary item from the auxiliary file process portion 304, the picture item from the video file process portion 305, and the sound item from the audio file process portion 306.

[0456]

That is, at Step S341, the header/footer generation portion 331 of the data synthesis portion 307 (FIG. 53) first generates a header and a footer for the standard AV multiplexing format file. Further, the header/footer generation portion 331 arranges the file-based metadata from the metadata file process portion 303 in the header's header metadata. In addition, at Step S341, the header/footer generation portion 331 supplies the header/footer adding portion 333 with the header and the footer that are obtained as mentioned above. The process proceeds to Step S342.

[0457]

That is, at Step S342, the multiplexing portion 332 multiplexes the system item output from the metadata file process portion 303, the auxiliary item output from the auxiliary file process portion 304, the picture item output from the video file process portion 305, and the sound item output from the audio file process portion 306, and supplies the resulting multiplexed edit unit sequence as a body to the header/footer adding portion 333. The process proceeds to Step S343.

[0458]

At Step S343, the header/footer adding portion 333 adds the header and the footer supplied from the header/footer generation portion 331 to the body supplied from the multiplexing portion 332. Accordingly, the header/footer adding portion 333 configures and outputs a standard AV multiplexing format file to terminate the synthesis process.

[0459]

As described abobe, efficient read and write processes can be provided by configuring a file so that it is sized to be an integral multiple of units of reading and writing on the recording medium.

[0460]

In addition, in a case where the second data to be placed at the beginning of the file and the third data to be placed at the end thereof are generated and added to the first, second, or third data to generate the fourth data so that the data amount of the first, second, or third data corresponds to an integral multiple of the unit of reading and writing on a recording medium, the usability of recording media improves, and the more efficient read and write processes can be provided when files are read from or written to recording media.

[0461]

The sequence of the above-mentioned processes may be implemented on the hardware or the software. When the sequence of processes is implemented on the software, programs constituting the software are installed on a general-purpose computer and the like.

[0462]

FIG. 59 shows an exemplary configuration of a computer installed with a program to implement the sequence of the above-mentioned processes.

[0463]

The program can be previously recorded on a hard disk 405 or ROM 403 as a recording medium contained in the computer.

[0464]

Alternatively, the program can be temporarily or permanently stored (recorded) on a removable recording medium 411 such as a flexible disk, a CD-ROM (Compact Disc Read Only Memory), an MO (Magneto Optical) disk, a DVD (Digital Versatile Disc), a magnetic disk, and a semiconductor memory. The removable recording medium 411 can be provided as so-called package software.

[0465]

In addition to installing the program on the computer from the removable recording medium 411 as mentioned above, the program can be wirelessly transferred to the computer from a download site via an artificial satellite for digital satellite broadcasting or wiredly transferred to the computer via networks such as LAN (Local Area Network) and the Internet. The computer can use a communication portion 408 to receive the transferred program and install it in the built-in hard disk 405.

[0466]

The computer includes a CPU (Central Processing Unit) 402. The CPU 402 connects with an input/output interface 410 via a bus 401. When a user inputs an instruction to the CPU 402 via the input/output interface 410 by operating an input portion 407, that is composed of a keyboard, a mouse, a microphone, and the like, the CPU 402 accordingly executes the program stored in the ROM (Read Only Memory) 403. Alternatively, the CPU 402 loads the program stored in the hard disk 405, the program transferred from a satellite or a network and received at the communication portion 408 to be installed on the hard disk 405, or the program read from the removable recording medium 411 mounted on the drive 409 and installed on the hard disk 405 into a RAM (Random Access Memory) 404 for execution. Accordingly, the CPU 402 performs the processes according to the above-mentioned flowcharts or the above-mentioned block diagram configurations. Depending on needs, for example, the CPU 402 outputs a process result from an output portion 406 composed of an LCD (Liquid Crystal Display), a speaker, and the like via the input/output interface 410, or transmits the process result from the communication portion 408 and records it on the hard disk 405, for example.

[0467]

In addition, the program may be processed on one computer or multiple computers in a distributed processing fashion. Further, the program may be transferred to a remote computer for execution.

[0468]

As mentioned above, since the standard AV multiplexing format file where the video data and the audio data are multiplexed and are arranged in the body and the AV independent format file where the video data or the audio data is collectively arranged in the body are mutually converted, the standard AV multiplexing format can be used when transmit files (exchange or stream files) via the network 4, while the AV independent format can be used when recording files on the optical disk 7, for example.

[0469]

When an AV independent format file is recorded on the optical disk 7, it becomes possible to easily perform the AV independent editing, for example.

[0470]

Further, since the AV independent format allows frame-based metadata to be collectively (integrally) arranged in a single file (frame-based metadata file), the frame-based metadata can be retrieved at high speed.

[0471]

Further, since the AV independent format uses the WAVE system as a coding system for audio data, it becomes possible to reduce the data amount of audio data compared to the standard AV multiplexing format in which the AES3 coding system is employed.

[0472]

In addition, in the AV independent format, the header, the body, and the footer having the same format as the standard AV multiplexing format are employed, and the header and the footer have the same format as the standard AV multiplexing format. Accordingly, a standard apparatus compliant with the standard AV multiplexing format can transmit and receive AV independent format files and read or write these files on recording media.

[0473]

Further, with respect to the standard AV multiplexing format file, the body contains multiple essences such as the video data, the audio data, the user data, and the frame-based metadata in a multiplexed fashion. By contrast, with respect to the AV independent format files (video files and audio files), only video data or audio data is placed in the body. Therefore, the AV independent format file can be an MXF file whose body is composed of a single essence. An apparatus that can interpret the MXF having the single-essence body can read the contents of a video file or an audio file that is an MXF file having the single-essence body.

[0474]

According to the embodiment, the disk apparatus 1 reads and writes AV independent format files on the optical disk 7. In addition, AV independent format files can be read from or written to not only disk-shaped recording media such as the optical disk 7, but also tape-shaped recording media such as magnetic tape, semiconductor memory, and the like.

[0475]

In addition, in the embodiment in FIG. 1, the single disk apparatus 1 is composed of the disk drive apparatus 11, the format conversion portion 12, and the communication I/F 13. However, the disk drive apparatus 11, the format conversion portion 12, and the communication I/F 13 may be independent apparatuses.

[0476]

Further, in the embodiment, MXF compliant files are employed as standard AV multiplexing format files. However, in addition to MXF compliant files, a file that is composed of the header, the body, and the footer, in which two (or more) pieces of any data are arranged in a multiplexed fashion may also be employed as the standard AV multiplexing files.

[0477]

Further, in the embodiment, the multiplexed video data and audio data is placed in the body of a standard AV multiplexing format file, but multiplexed (stream of) two or more pieces of video data or multiplexed (stream of) two or more pieces of audio data, for example may also be placed in the standard AV multiplexing format file's body.

[0478]

Effect of the Invention

As mentioned above, according to the first invention, it becomes possible to perform efficient read and write processes.

[0479]

Further, according to the first invention, it is possible to improve the usability of recording media and perform efficient read and write processes in reading or writing files from/to recording media.

[0480]

According to the second invention, it becomes possible to perform efficient read and write processes.

[0481]

Further, according to the second invention, it becomes possible to improve the usability and perform efficient read and write processes in reading or writing files from/to recording media.

[BRIEF DESCRIPTION OF THE DRAWINGS]

[FIG. 1] A block diagram showing an exemplary configuration of an AV network system according to an embodiment of the present invention.

[FIG. 2] A diagram showing a standard AV multiplexing format.

[FIG. 3] A diagram showing an AV independent format.

[FIG. 4] A diagram showing the AV independent format.

[FIG. 5] A diagram showing the AV independent format.

[FIG. 6] A diagram showing the AV independent format.

[FIG. 7] A block diagram showing an exemplary configuration of a format conversion portion 12.

[FIG. 8] A block diagram showing an exemplary configuration of a standard/independent format conversion portion 21.

[FIG. 9] A block diagram showing an exemplary configuration of a video file generation portion 41.

[FIG. 10] A block diagram showing an exemplary configuration of an audio file generation portion 43.

[FIG. 11] A flowchart illustrating a master file generation process.

[FIG. 12] A flowchart illustrating a metadata file generation process in units of files.

[FIG. 13] A flowchart illustrating a metadata file generation process in units of frames.

[FIG. 14] A flowchart illustrating an auxiliary file generation process.

[FIG. 15] A flowchart illustrating a video file generation process.

[FIG. 16] A flowchart illustrating an audio file generation process.

[FIG. 17] A block diagram showing an exemplary configuration of a disk drive apparatus 11.

[FIG. 18] A block diagram showing an exemplary configuration of a data conversion portion 118.

[FIG. 19] A diagram illustrating a structure of a low resolution data file.

[FIG. 20] A diagram illustrating a structure of a low resolution data file.

[FIG. 21] A diagram illustrating a structure of a system item.

[FIG. 22] A diagram illustrating a structure of a picture essence.

[FIG. 23] A diagram illustrating the data amount of a KLV-structured picture essence.

[FIG. 24] A diagram showing a configuration of a sound essence.

[FIG. 25]A block diagram showing a configuration of a low resolution data generation portion 142.

[FIG. 26] A block diagram illustrating a configuration of a video file process portion 164.

[FIG. 27] A block diagram illustrating a configuration of an audio file process portion 165.

[FIG. 28] A block diagram showing a configuration of a data synthesis portion 166.

[FIG. 29] A flowchart illustrating a video file process.

[FIG. 30] A flowchart illustrating an audio file process.

[FIG. 31] A flowchart illustrating a metadata file process.

[FIG. 32] A flowchart showing a low resolution file synthesis process.

[FIG. 33] A flowchart showing a recording process by means of a control portion 119.

[FIG. 34] A flowchart illustrating an audio data recording task.

[FIG. 35] A diagram showing changes in total data amount La of audio data and total data amount Lv of video data.

[FIG. 36] A diagram showing states of recording audio data and video data on an optical disk 7.

[FIG. 37] A flowchart illustrating a video data recording task.

[FIG. 38] A diagram showing changes in total data amount La of audio data and total data amount Lv of video data.

[FIG. 39] A flowchart illustrating a low resolution data recording task.

[FIG. 40] A flowchart illustrating a metadata recording task.

[FIG. 41] A diagram showing the total data amount of data recorded in a memory 18.

[FIG. 42] A diagram showing the total data amount of data recorded in the memory 18.

[FIG. 43] A diagram showing the total data amount of data recorded in the memory 18.

[FIG. 44] A diagram showing the total data amount of data recorded in the memory 18.

[FIG. 45] A diagram showing the total data amount of data recorded in the memory 18.

[FIG. 46] A diagram showing states of recording data on the optical disk 7.

[FIG. 47] A diagram showing states of recording data on the optical disk 7.

[FIG. 48] Diagrams illustrating data recorded on the optical disk 7.

[FIG. 49] A flowchart illustrating a recording process.

[FIG. 50] A block diagram showing an exemplary configuration of an independent/standard format conversion portion 22.

[FIG. 51] A block diagram showing an exemplary configuration of a video file process portion 305.

[FIG. 52] A block diagram showing an exemplary configuration of an audio file process portion 306.

[FIG. 53] A block diagram showing an exemplary configuration of a data synthesis portion 307.

[FIG. 54] A flowchart illustrating a metadata file process.

[FIG. 55] A flowchart illustrating an auxiliary file process.

[FIG. 56] A flowchart illustrating a video file process.

[FIG. 57] A flowchart illustrating an audio file process.

[FIG. 58] A flowchart illustrating a synthesis process.

[FIG. 59] A block diagram showing an exemplary configuration of a computer to which an embodiment of the present invention is applied.

[DESCRIPTION OF REFERENCE NUMERALS]

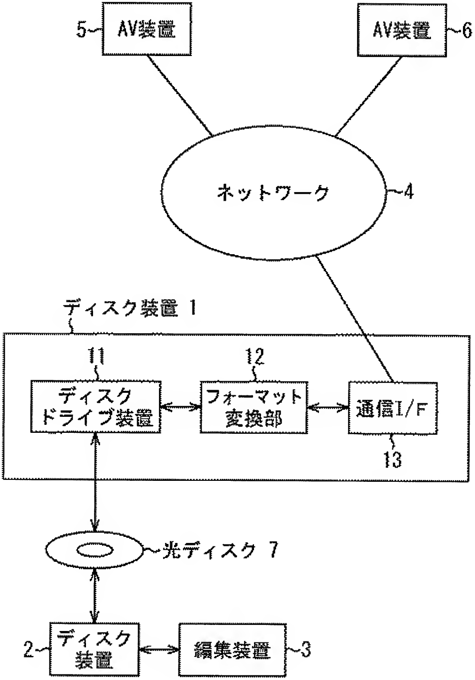
[0053]

1, 2 disk apparatus, 3 editing apparatus, 4 Network, 5, 6 AV apparatus, 7 optical disk, 11 disk drive apparatus, 12 format conversion portion, 13 communication I/F 21 standard/independent format conversion portion, 22 independent/standard format conversion portion, 31 buffer, 32 master file generation portion, 33 header acquisition portion, 34 body acquisition portion, 35 header metadata extraction portion, 36 system item extraction portion, 37 metadata file generation portion, 38 auxiliary item extraction portion, 39 auxiliary file generation portion, 40 picture item extraction portion, 41 video file generation portion, 42 sound item extraction portion, 43 audio file generation portion, 44 buffer, 51 connection portion, 52 footer generation portion, 53 header generation portion, 54 filler generation portion, 55 KLV encoder, 61 KLV decoder, 62 channel separation portion, 63 data conversion portion, 64 KLV encoder, 65 header generation portion, 66 footer generation portion, 67 filler generation portion, 68 KLV encoder, 71 beginning data generation portion, 111 spindle motor, 112 pickup portion, 113 RF amplifier, 114 servo control portion, 115 signal process portion, 116 memory controller, 117 memory, 118 data conversion portion, 119 control portion, 120 operation portion, 141 data amount detection portion, 142 low resolution data generation portion, 161 buffer, 162 file process portion, 163 metadata process portion, 164 video file process portion, 165 audio file process portion, 166 data synthesis portion, 167 buffer, 181 decomposition portion, 182 data conversion portion, 203 channel multiplexing portion, 204 KLV encoder, 205 filler generation portion, 206 KLV encoder, 221 multiplexing portion, 222 footer generation portion, 223 header generation portion, 224 filler generation portion, 301 buffer, 302 file acquisition portion, 303 metadata file process portion, 304 auxiliary file process portion, 305 video file generation portion, 306 audio file process portion, 307 data synthesis portion, 308 buffer, 311 header/footer removal portion, 312 decomposition portion, 321 header/footer removal portion, 322 KLV decoder, 323 data conversion portion, 324 channel multiplexing portion, 325 KLV encoder, 331 header/footer generation portion, 332 multiplexing portion, 333 header/footer adding portion, 401 bus, 402 CPU, 403 ROM, 404 RAM, 405 hard disk, 406 output portion, 407 input portion, 408 communication portion, 409 drive, 410 input/output interface, 411 removable recording medium

【書類名】図面

【図１】

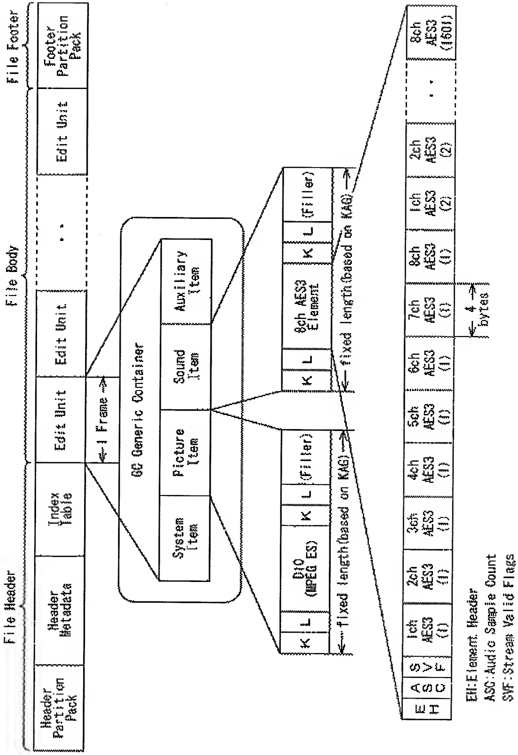
図1



AV装置
5
AV装置
6
ネットワーク
4
ディスク装置 1
11
12
ディスク
ドライブ装置
フォーマット
変換部
通信I/F
13
光ディスク 7
ディスク
装置
2
編集装置
3

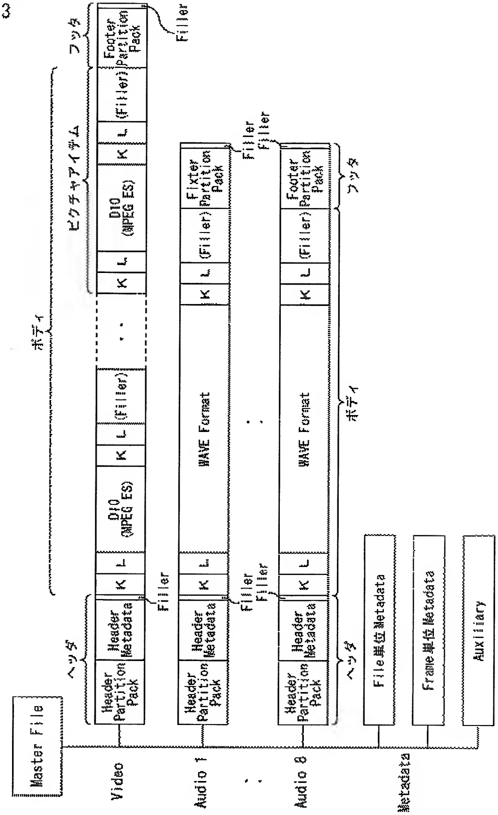
【図２】

図2


File Header
File Body
File Footer
Header Partition Pack
Header Metadata
Index Table
Edit Unit
Edit Unit
Edit Unit
Edit Unit
Footer Partition Pack
1 Frame
GC Generic Container
System Item
Picture Item
Sound Item
Auxiliary Item
K
L
D10 (MPEG ES)
K
L
(Filler)
fixed length (based on KAG)
K
L
8ch AES3 Element
K
L
(Filler)
fixed length (based on KAG)
EH
ASC
SVF
1ch AES3 (1)
2ch AES3 (1)
3ch AES3 (1)
4ch AES3 (1)
5ch AES3 (1)
6ch AES3 (1)
7ch AES3 (1)
8ch AES3 (1)
1ch AES3 (2)
2ch AES3 (2)
8ch AES3 (1601)
4 bytes
EH: Element Header
ASC: Audio Sample Count
SVF: Stream Valid Flags

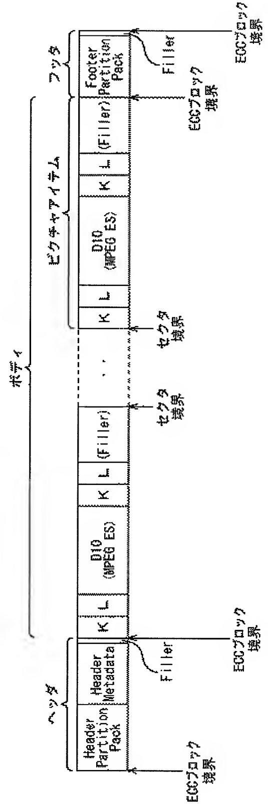
【図３】

図3

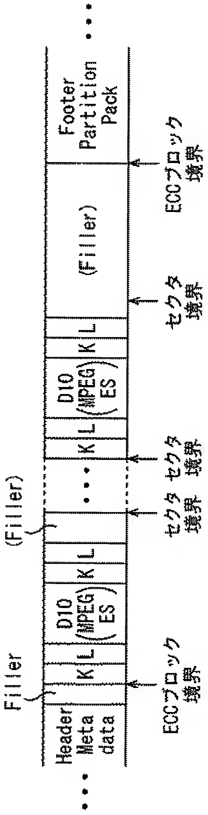

Master File
Video
Audio 1
・・
Audio 8
Metadata
ヘッダ
Header Partition Pack
Header Metadata
Filler
K
L
DIO (MPEG ES)
K
L
(Filler)
・・
K
L
DIO (MPEG ES)
K
L
(Filler)
ピクチャアイテム
ボディ
Footer Partition Pack
フッタ
Filler
Header Partition Pack
Header Metadata
Filler
K
L
WAVE Format
K
L
(Filler)
Fixter Partition Pack
Filler
Header Partition Pack
Header Metadata
Filler
K
L
WAVE Format
K
L
(Filler)
Footer Partition Pack
Filler
ヘッダ
ボディ
フッタ
File単位Metadata
Frame単位Metadata
Auxiliary

【図４】

図4

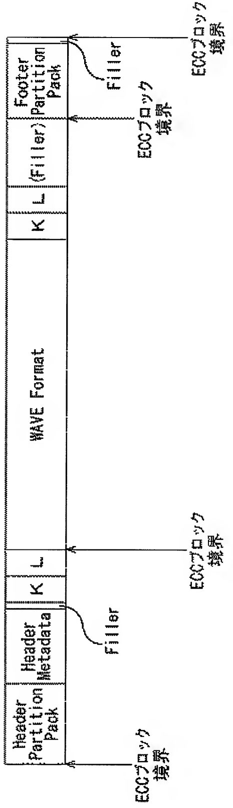

ヘッダ
Header Partition Pack
Header Metadata
Filler
ECCブロック境界
ECCブロック境界
ボディ
K
L
D10 (MPEG ES)
K
L
(Filler)
セクタ境界
セクタ境界
ピクチャアイテム
K
L
D10 (MPEG ES)
K
L
(Filler)
ECCブロック境界
フッタ
Footer Partition Pack
Filler
ECCブロック境界

【図５】
図5

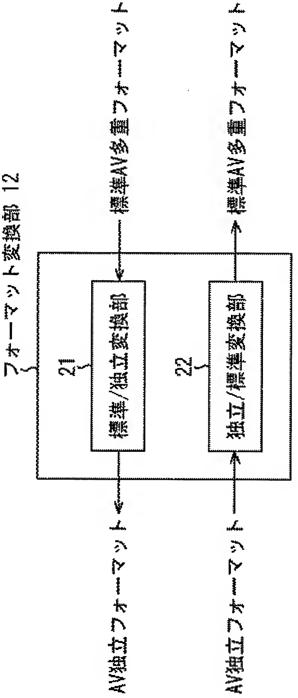

Header Meta data
Filler
K
L
D10 (MPEG ES)
K
L
(Filler)
K
L
D10 (MPEG ES)
K
L
(Filler)
Footer Partition Pack
ECCブロック境界
セクタ境界
セクタ境界
セクタ境界
ECCブロック境界

【図６】

図6

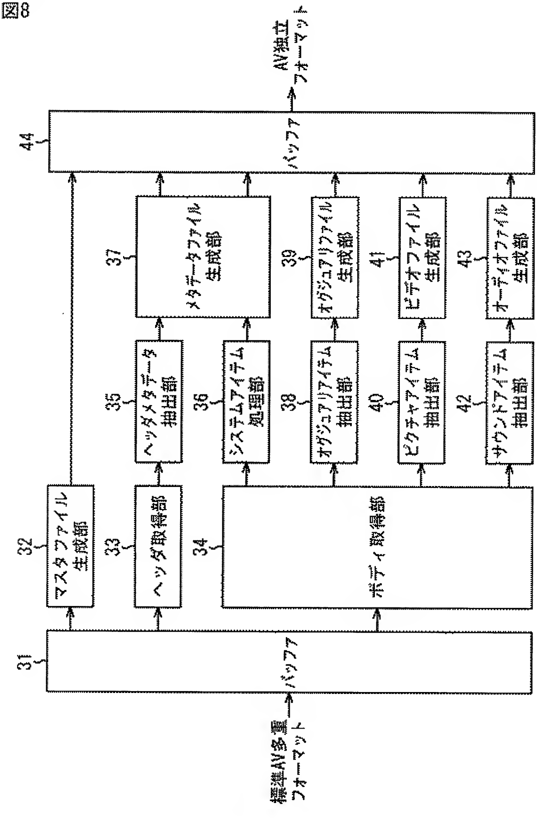

ECCブロック境界
Header Partition Pack
Header Metadata
Filler
K
L
ECCブロック境界
WAVE Format
K
L
(Filler)
ECCブロック境界
Footer Partition Pack
Filler
ECCブロック境界

【図７】
図7


フォーマット変換部 12
21
標準/独立変換部
22
独立/標準変換部
標準AV多重フォーマット
標準AV多重フォーマット
AV独立フォーマット
AV独立フォーマット

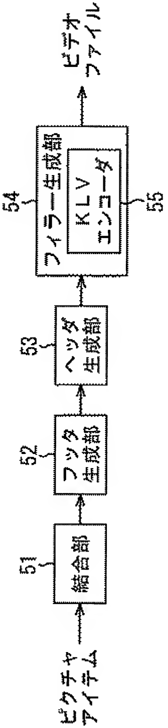
【図８】

図8


標準AV多重フォーマット
31
バッファ
32
マスタファイル生成部
33
ヘッダ取得部
34
ボディ取得部
35
ヘッダメタデータ抽出部
36
システムアイテム処理部
37
メタデータファイル生成部
38
オグジュアリアイテム抽出部
39
オグジュアリファイル生成部
40
ピクチャアイテム抽出部
41
ビデオファイル生成部
42
サウンドアイテム抽出部
43
オーディオファイル生成部
44
バッファ
AV独立フォーマット

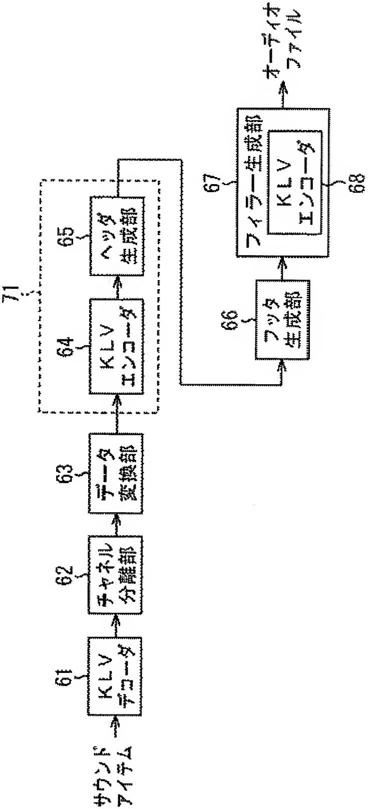
【図９】

図9


ピクチャ
アイテム
51
結合部
52
フッタ
生成部
53
ヘッダ
生成部
54
フィラー生成部
KLV
エンコーダ
55
ビデオ
ファイル

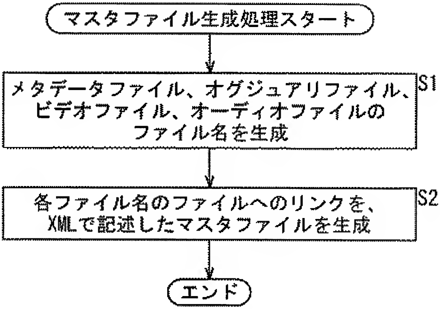
【図１０】

図10


サウンドアイテム
61
KLVデコーダ
62
チャネル分離部
63
データ変換部
71
64
KLVエンコーダ
65
ヘッダ生成部
66
フッタ生成部
67
フィラー生成部
68
KLVエンコーダ
オーディオファイル

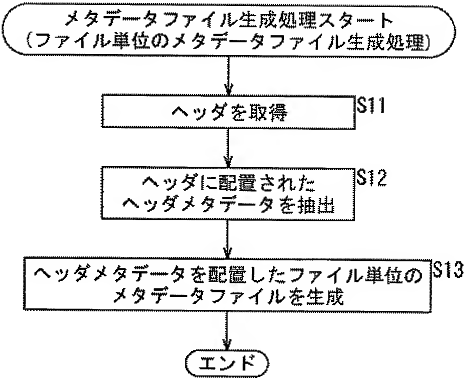
【図１１】

図11


マスタファイル生成処理スタート
メタデータファイル、オグジュアリファイル、ビデオファイル、オーディオファイルのファイル名を生成
S1
各ファイル名のファイルへのリンクを、XMLで記述したマスタファイルを生成
S2
エンド

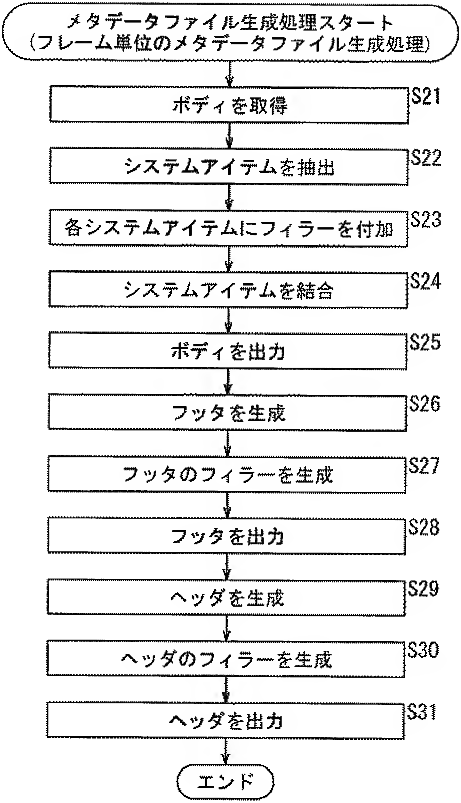
【図１２】

図12


メタデータファイル生成処理スタート
（ファイル単位のメタデータファイル生成処理）
ヘッダを取得
S11
ヘッダに配置された
ヘッダメタデータを抽出
S12
ヘッダメタデータを配置したファイル単位の
メタデータファイルを生成
S13
エンド

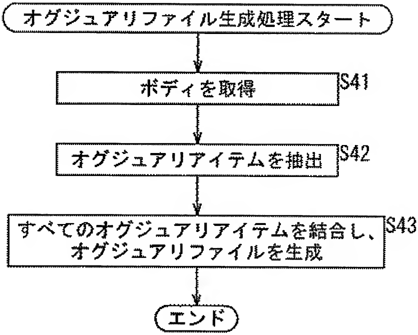
【図１３】

図13


メタデータファイル生成処理スタート
(フレーム単位のメタデータファイル生成処理)
ボディを取得
S21
システムアイテムを抽出
S22
各システムアイテムにフィラーを付加
S23
システムアイテムを結合
S24
ボディを出力
S25
フッタを生成
S26
フッタのフィラーを生成
S27
フッタを出力
S28
ヘッダを生成
S29
ヘッダのフィラーを生成
S30
ヘッダを出力
S31
エンド

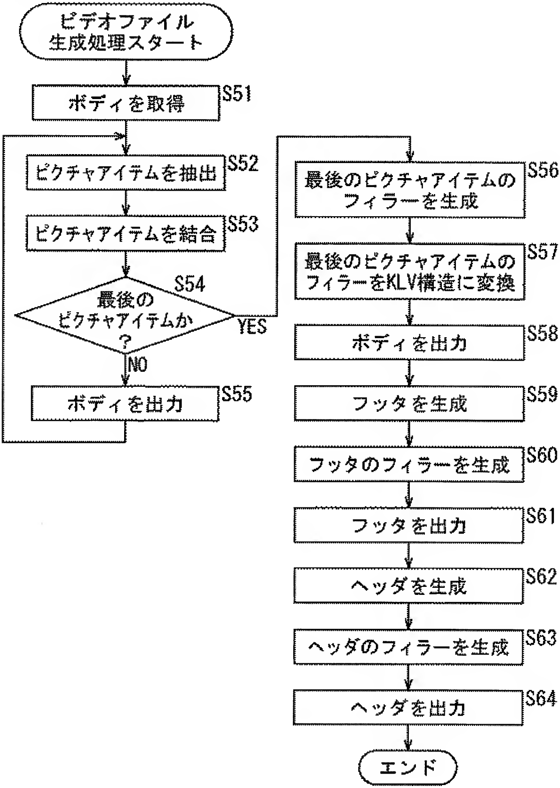
【図１４】

図14


オグジュアリファイル生成処理スタート
ボディを取得
S41
オグジュアリアイテムを抽出
S42
すべてのオグジュアリアイテムを結合し、オグジュアリファイルを生成
S43
エンド

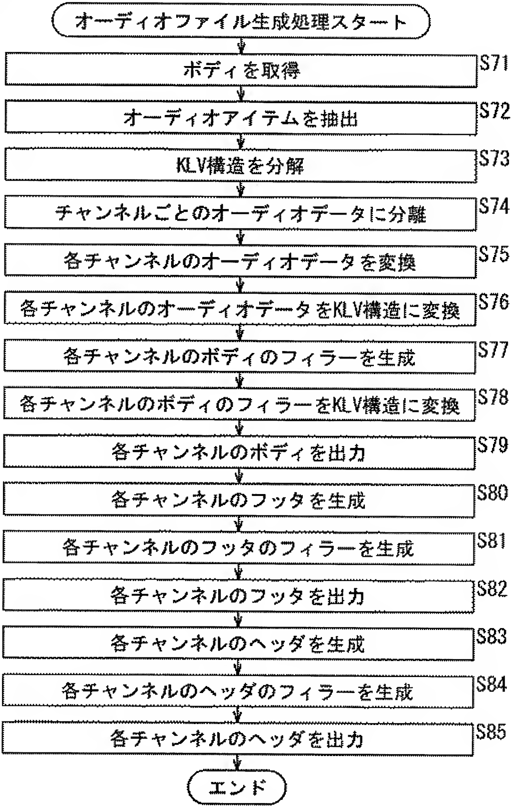
【図１５】

図15


ビデオファイル
生成処理スタート
ボディを取得
S51
ピクチャアイテムを抽出
S52
ピクチャアイテムを結合
S53
最後の
ピクチャアイテムか
？
S54
YES
NO
ボディを出力
S55
最後のピクチャアイテムの
フィラーを生成
S56
最後のピクチャアイテムの
フィラーをKLV構造に変換
S57
ボディを出力
S58
フッタを生成
S59
フッタのフィラーを生成
S60
フッタを出力
S61
ヘッダを生成
S62
ヘッダのフィラーを生成
S63
ヘッダを出力
S64
エンド

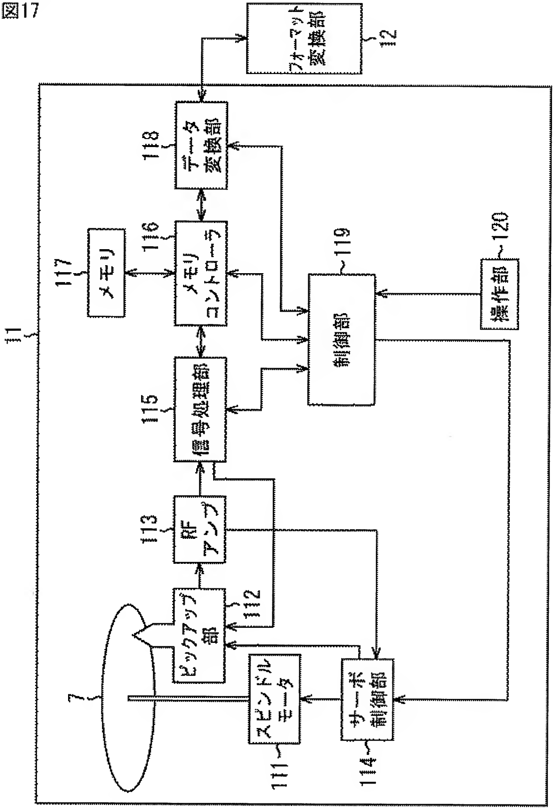
【図１６】

図16


オーディオファイル生成処理スタート
ボディを取得
S71
オーディオアイテムを抽出
S72
KLV構造を分解
S73
チャンネルごとのオーディオデータに分離
S74
各チャンネルのオーディオデータを変換
S75
各チャンネルのオーディオデータをKLV構造に変換
S76
各チャンネルのボディのフィラーを生成
S77
各チャンネルのボディのフィラーをKLV構造に変換
S78
各チャンネルのボディを出力
S79
各チャンネルのフッタを生成
S80
各チャンネルのフッタのフィラーを生成
S81
各チャンネルのフッタを出力
S82
各チャンネルのヘッダを生成
S83
各チャンネルのヘッダのフィラーを生成
S84
各チャンネルのヘッダを出力
S85
エンド

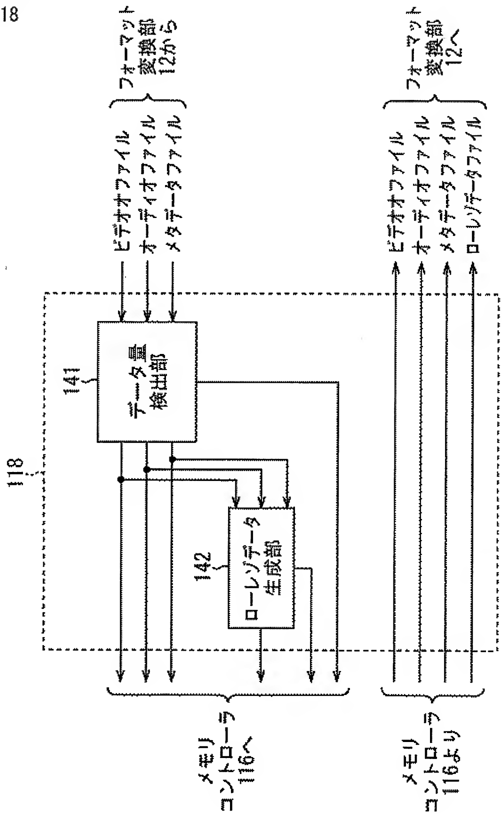
【図１７】

図17

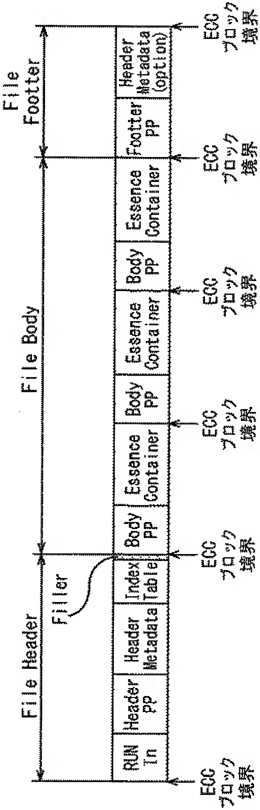

フォーマット変換部
12
11
データ変換部
118
メモリ
117
メモリコントローラ
116
信号処理部
115
制御部
119
操作部
120
RFアンプ
113
ピックアップ部
112
スピンドルモータ
111
サーボ制御部
114
7

【図１８】

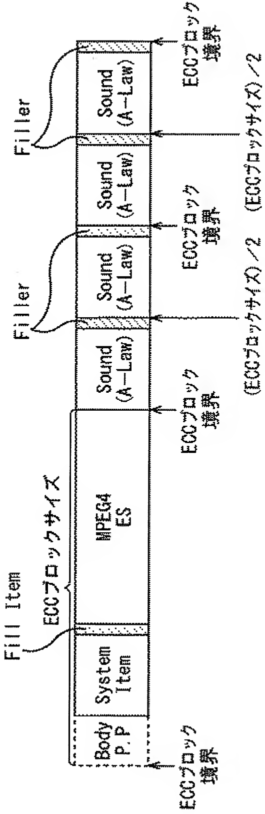
図18


フォーマット変換部12から
ビデオファイル
オーディオファイル
メタデータファイル
141
データ量検出部
118
142
ローレゾデータ生成部
メモリコントローラ116へ
フォーマット変換部12へ
ビデオファイル
オーディオファイル
メタデータファイル
ローレゾデータファイル
メモリコントローラ116より

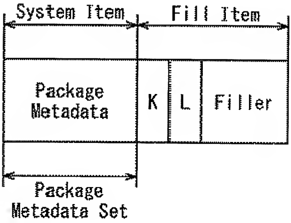
【図１９】
図19


File Header
Filler
File Body
File Footter
RUN In
Header PP
Header Metadata
Index Table
Body PP
Essence Container
Body PP
Essence Container
Body PP
Essence Container
Footter PP
Header Metadata (option)
ECC ブロック境界
ECC ブロック境界
ECC ブロック境界
ECC ブロック境界
ECC ブロック境界
ECC ブロック境界

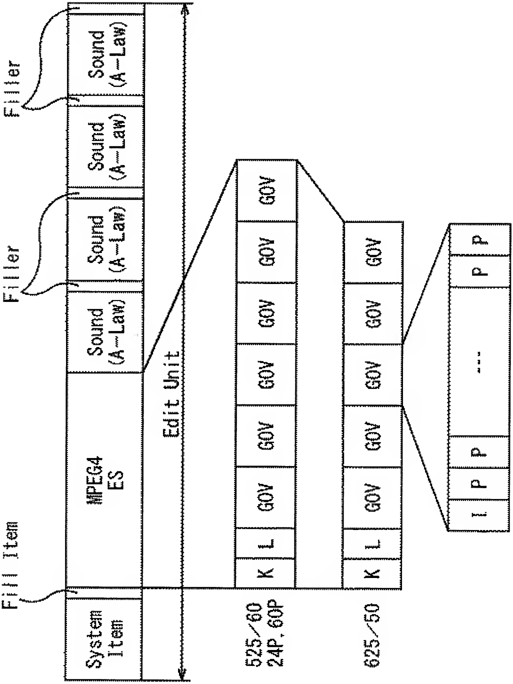
【図２０】
図20


Body P.P
System Item
Fill Item
ECCブロックサイズ
MPEG4 ES
Sound (A−Law)
Filler
Sound (A−Law)
Sound (A−Law)
Filler
Sound (A−Law)
ECCブロック境界
ECCブロック境界
(ECCブロックサイズ)／2
ECCブロック境界
(ECCブロックサイズ)／2
ECCブロック境界

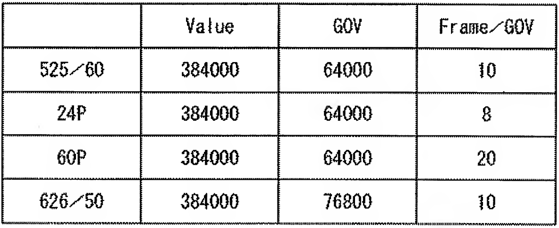
【図２１】
図21


System Item
Fill Item
Package Metadata
K
L
Filler
Package Metadata Set

【図２２】
図22

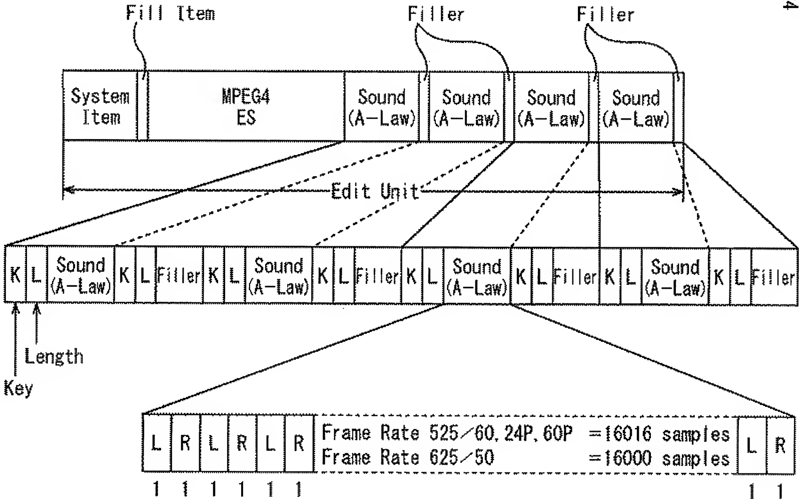

Fill Item
Filler
Filler
System Item
MPEG4 ES
Sound (A−Law)
Sound (A−Law)
Sound (A−Law)
Sound (A−Law)
Edit Unit
525／60 24P, 60P
K
L
GOV
GOV
GOV
GOV
GOV
GOV
625／50
K
L
GOV
GOV
GOV
GOV
GOV
I
P
P
…
P
P

【図２３】
図23

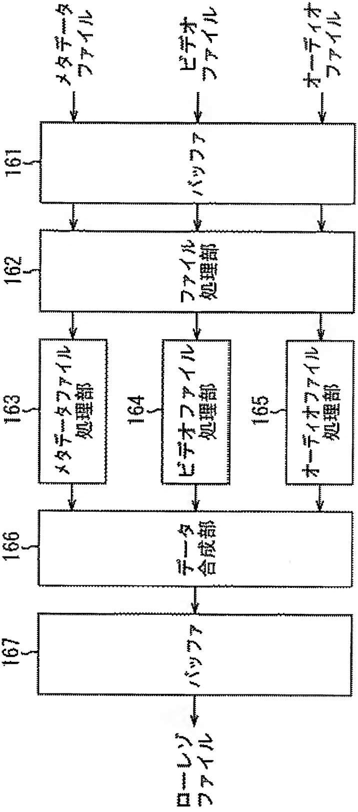


	Value	GOV	Frame／GOV
525／60	384000	64000	10
24P	384000	64000	8
60P	384000	64000	20
626／50	384000	76800	10

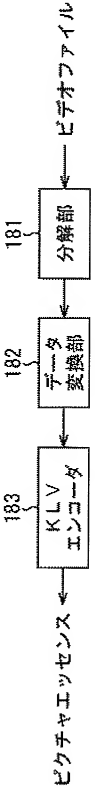
【図２４】
図24


Fill Item
Filler
Filler
System Item
MPEG4 ES
Sound (A-Law)
Sound (A-Law)
Sound (A-Law)
Sound (A-Law)
Edit Unit
K
L
Sound (A-Law)
K
L
Filler
K
L
Sound (A-Law)
K
L
Filler
K
L
Sound (A-Law)
K
L
Filler
K
L
Sound (A-Law)
K
L
Filler
Length
Key
L
R
L
R
L
R
Frame Rate 525／60, 24P, 60P =16016 samples
Frame Rate 625／50 =16000 samples
L
R
1
1
1
1
1
1
1
1

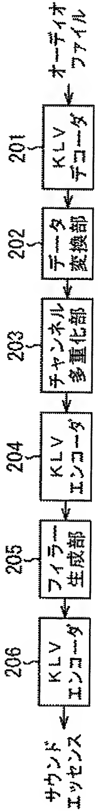
【図２５】
図25


メタデータ
ファイル
ビデオ
ファイル
オーディオ
ファイル
161
バッファ
162
ファイル
処理部
163
メタデータファイル
処理部
164
ビデオファイル
処理部
165
オーディオファイル
処理部
166
データ
合成部
167
バッファ
ローレゾ
ファイル

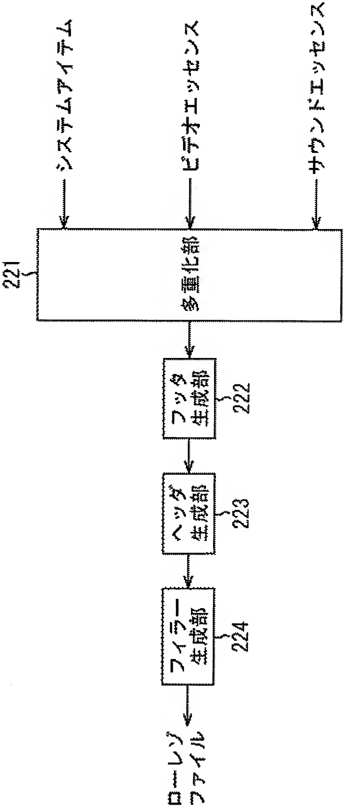
【図２６】
図26



【図２７】
図27

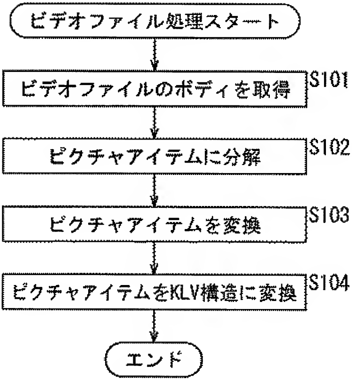

オーディオファイル
201
KLV デコーダ
202
データ変換部
203
チャンネル多重化部
204
KLV エンコーダ
205
フィラー生成部
206
KLV エンコーダ
サウンドエッセンス

【図２８】
図28


システムアイテム
ビデオエッセンス
サウンドエッセンス
多重化部
221
フッタ生成部
222
ヘッダ生成部
223
フィラー生成部
224
ローレゾファイル

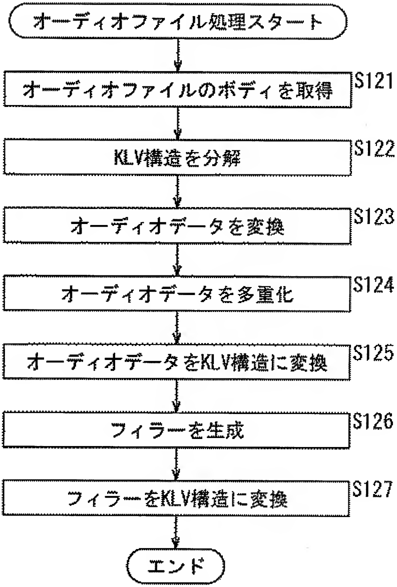
【図２９】

図29


ビデオファイル処理スタート
ビデオファイルのボディを取得
S101
ピクチャアイテムに分解
S102
ピクチャアイテムを変換
S103
ピクチャアイテムをKLV構造に変換
S104
エンド

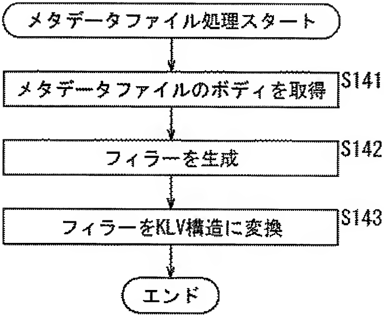
【図３０】

図30


オーディオファイル処理スタート
オーディオファイルのボディを取得
S121
KLV構造を分解
S122
オーディオデータを変換
S123
オーディオデータを多重化
S124
オーディオデータをKLV構造に変換
S125
フィラーを生成
S126
フィラーをKLV構造に変換
S127
エンド

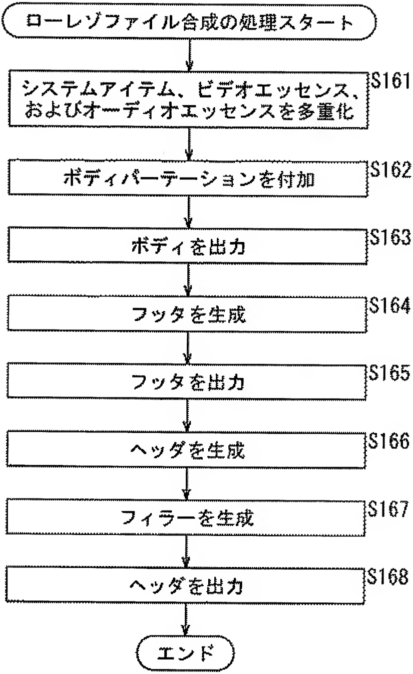
【図３１】

図31


メタデータファイル処理スタート
メタデータファイルのボディを取得
S141
フィラーを生成
S142
フィラーをKLV構造に変換
S143
エンド

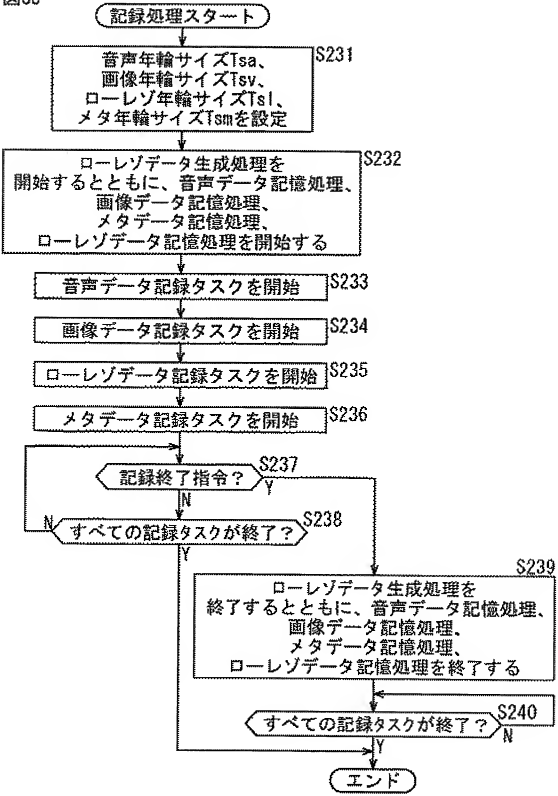
【図３２】

図32


ローレゾファイル合成の処理スタート
システムアイテム、ビデオエッセンス、およびオーディオエッセンスを多重化
S161
ボディパーテーションを付加
S162
ボディを出力
S163
フッタを生成
S164
フッタを出力
S165
ヘッダを生成
S166
フィラーを生成
S167
ヘッダを出力
S168
エンド

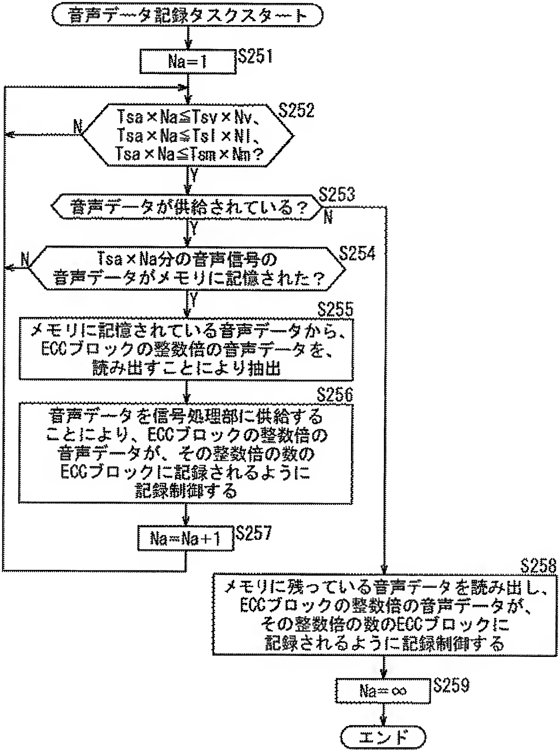
【図３３】

図33

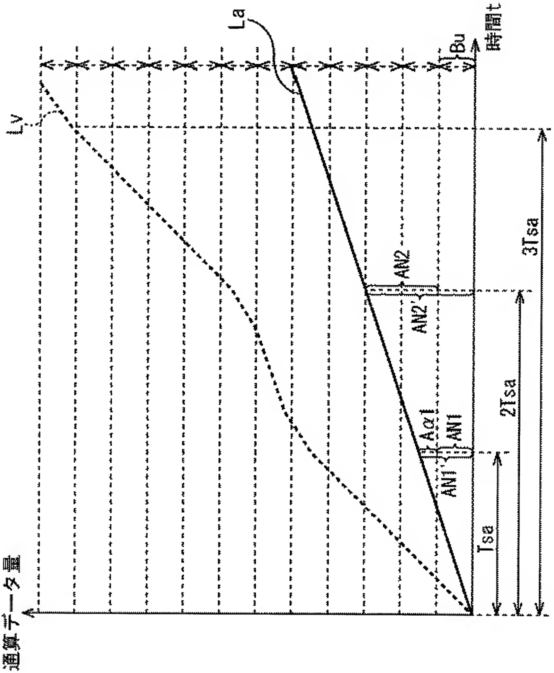

記録処理スタート
音声年輪サイズTsa、
画像年輪サイズTsv、
ローレゾ年輪サイズTsl、
メタ年輪サイズTsmを設定
S231
ローレゾデータ生成処理を
開始するとともに、音声データ記憶処理、
画像データ記憶処理、
メタデータ記憶処理、
ローレゾデータ記憶処理を開始する
S232
音声データ記録タスクを開始
S233
画像データ記録タスクを開始
S234
ローレゾデータ記録タスクを開始
S235
メタデータ記録タスクを開始
S236
記録終了指令？
S237
Y
N
すべての記録タスクが終了？
S238
N
Y
S239
ローレゾデータ生成処理を
終了するとともに、音声データ記憶処理、
画像データ記憶処理、
メタデータ記憶処理、
ローレゾデータ記憶処理を終了する
すべての記録タスクが終了？
S240
N
Y
エンド

【図３４】

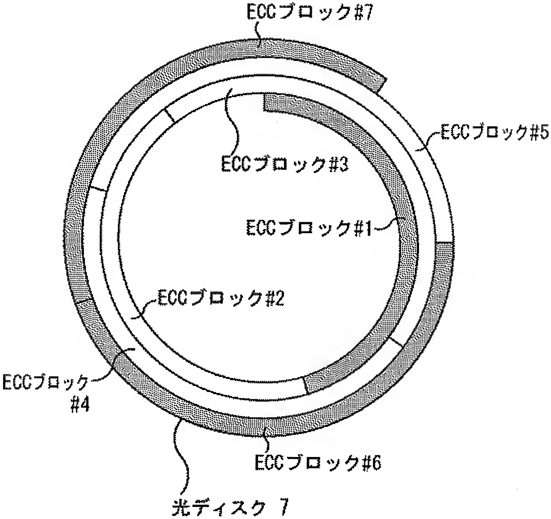
図34


音声データ記録タスクスタート
Na=1
S251
Tsa×Na≦Tsv×Nv、
Tsa×Na≦Tsl×Nl、
Tsa×Na≦Tsm×Nm？
S252
N
Y
音声データが供給されている？
S253
N
Y
Tsa×Na分の音声信号の
音声データがメモリに記憶された？
S254
N
Y
メモリに記憶されている音声データから、
ECCブロックの整数倍の音声データを、
読み出すことにより抽出
S255
音声データを信号処理部に供給する
ことにより、ECCブロックの整数倍の
音声データが、その整数倍の数の
ECCブロックに記録されるように
記録制御する
S256
Na=Na+1
S257
メモリに残っている音声データを読み出し、
ECCブロックの整数倍の音声データが、
その整数倍の数のECCブロックに
記録されるように記録制御する
S258
Na=∞
S259
エンド

【図３５】
図35



【図３６】
図36


ECCブロック#7
ECCブロック#5
ECCブロック#3
ECCブロック#1
ECCブロック#2
ECCブロック
#4
ECCブロック#6
光ディスク 7

【図３７】

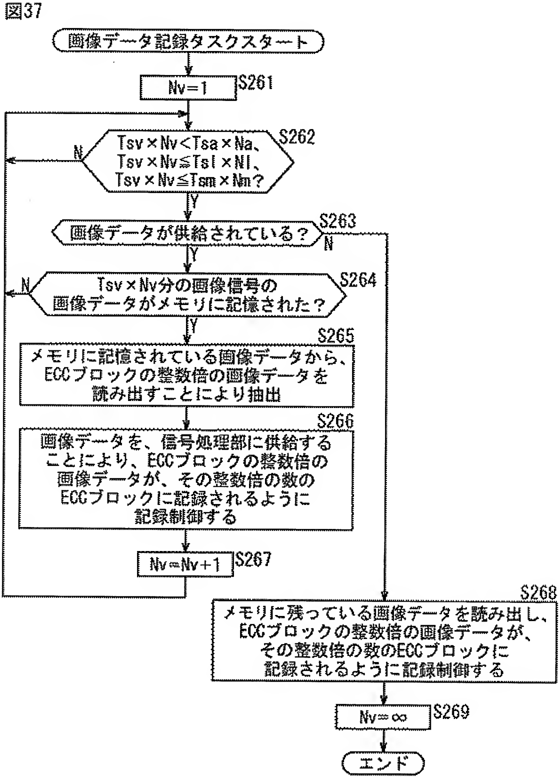
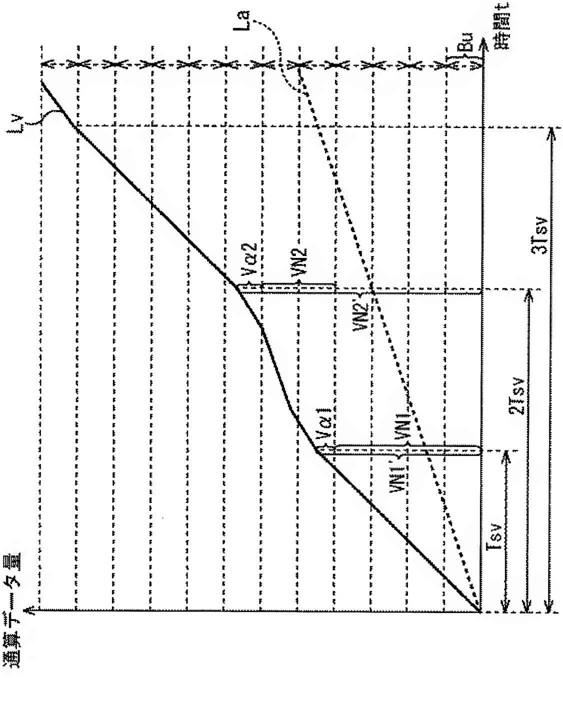


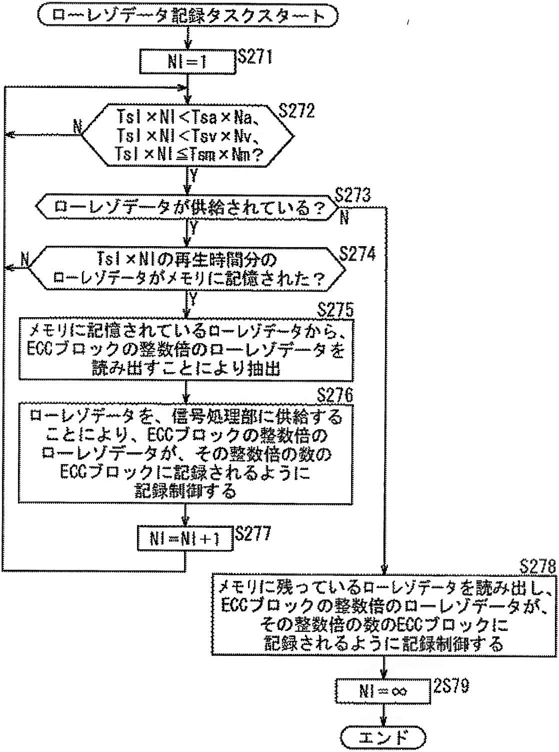
図37
画像データ記録タスクスタート
Nv=1
S261
Tsv×Nv<Tsa×Na、
Tsv×Nv≦Tsl×Nl、
Tsv×Nv≦Tsm×Nm？
S262
N
Y
画像データが供給されている？
S263
N
Y
Tsv×Nv分の画像信号の
画像データがメモリに記憶された？
S264
N
Y
メモリに記憶されている画像データから、
ECCブロックの整数倍の画像データを
読み出すことにより抽出
S265
画像データを、信号処理部に供給する
ことにより、ECCブロックの整数倍の
画像データが、その整数倍の数の
ECCブロックに記録されるように
記録制御する
S266
Nv=Nv+1
S267
メモリに残っている画像データを読み出し、
ECCブロックの整数倍の画像データが、
その整数倍の数のECCブロックに
記録されるように記録制御する
S268
Nv=∞
S269
エンド

【図３８】
図38



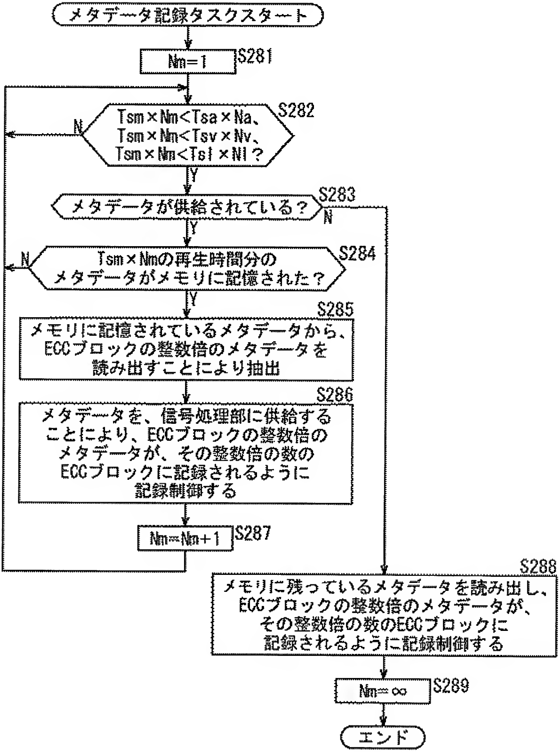
【図３９】

図39


ローレゾデータ記録タスクスタート
NI=1
S271
TsI×NI<Tsa×Na、
TsI×NI<Tsv×Nv、
TsI×NI≦Tsm×Nm？
S272
N
Y
ローレゾデータが供給されている？
S273
N
Y
TsI×NIの再生時間分の
ローレゾデータがメモリに記憶された？
S274
N
Y
S275
メモリに記憶されているローレゾデータから、
ECCブロックの整数倍のローレゾデータを
読み出すことにより抽出
S276
ローレゾデータを、信号処理部に供給する
ことにより、ECCブロックの整数倍の
ローレゾデータが、その整数倍の数の
ECCブロックに記録されるように
記録制御する
NI=NI+1
S277
S278
メモリに残っているローレゾデータを読み出し、
ECCブロックの整数倍のローレゾデータが、
その整数倍の数のECCブロックに
記録されるように記録制御する
NI=∞
2S79
エンド

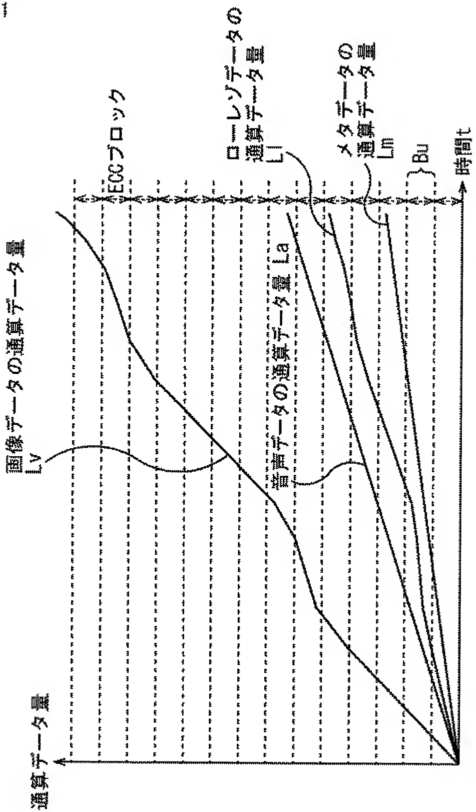
【図４０】

図40


メタデータ記録タスクスタート
Nm=1
S281
Tsm×Nm<Tsa×Na、
Tsm×Nm<Tsv×Nv、
Tsm×Nm<Tsl×Nl？
S282
N
Y
メタデータが供給されている？
S283
N
Y
Tsm×Nmの再生時間分の
メタデータがメモリに記憶された？
S284
N
Y
S285
メモリに記憶されているメタデータから、
ECCブロックの整数倍のメタデータを
読み出すことにより抽出
S286
メタデータを、信号処理部に供給する
ことにより、ECCブロックの整数倍の
メタデータが、その整数倍の数の
ECCブロックに記録されるように
記録制御する
Nm=Nm+1
S287
S288
メモリに残っているメタデータを読み出し、
ECCブロックの整数倍のメタデータが、
その整数倍の数のECCブロックに
記録されるように記録制御する
Nm=∞
S289
エンド

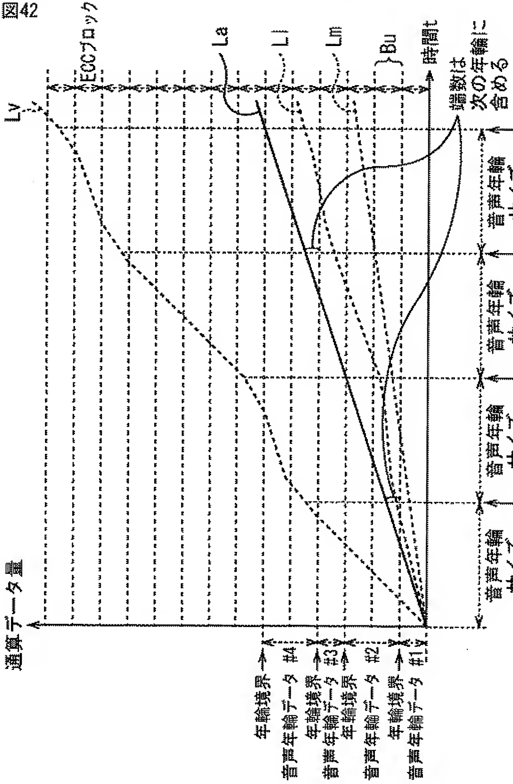
【図４１】

図41


画像データの通算データ量
Lv
ECCブロック
ローレンジデータの
通算データ量
Ll
メタデータの
通算データ量
Lm
Bu
時間t
音声データの通算データ量 La
通算データ量

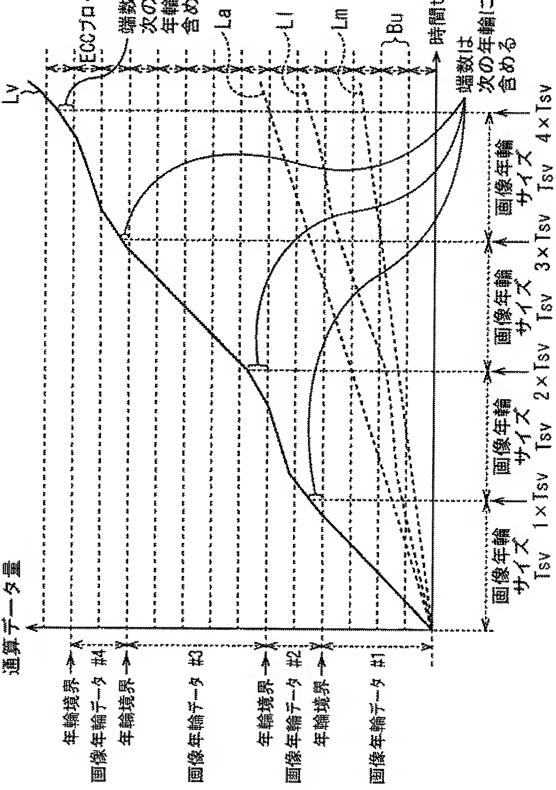
【図４２】

図42

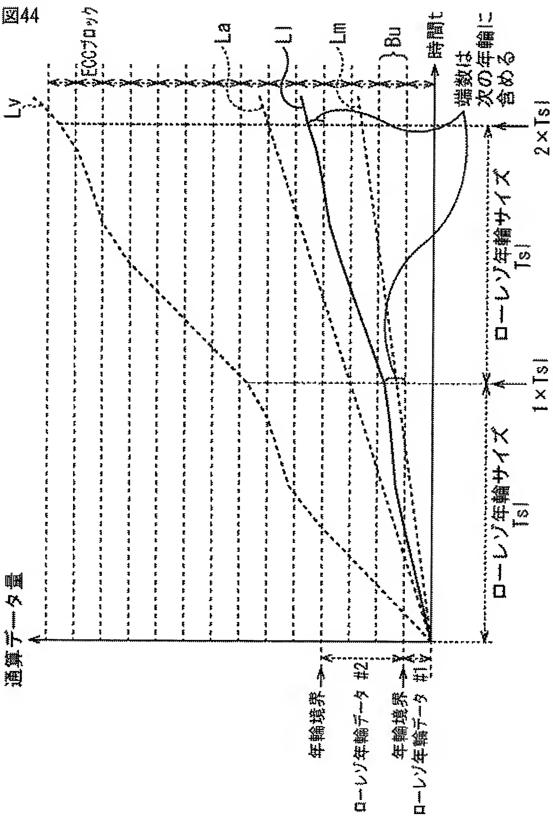

通算データ量
時間t
ECCブロック
Lv
La
Ll
Lm
Bu
端数は次の年輪に含める
音声年輪サイズ
Tsa
1×Tsa
音声年輪サイズ
Tsa
2×Tsa
音声年輪サイズ
Tsa
3×Tsa
音声年輪サイズ
Tsa
4×Tsa
年輪境界
音声年輪データ
#4
年輪境界
音声年輪データ
#3
年輪境界
音声年輪データ
#2
年輪境界
音声年輪データ
#1

【図４３】

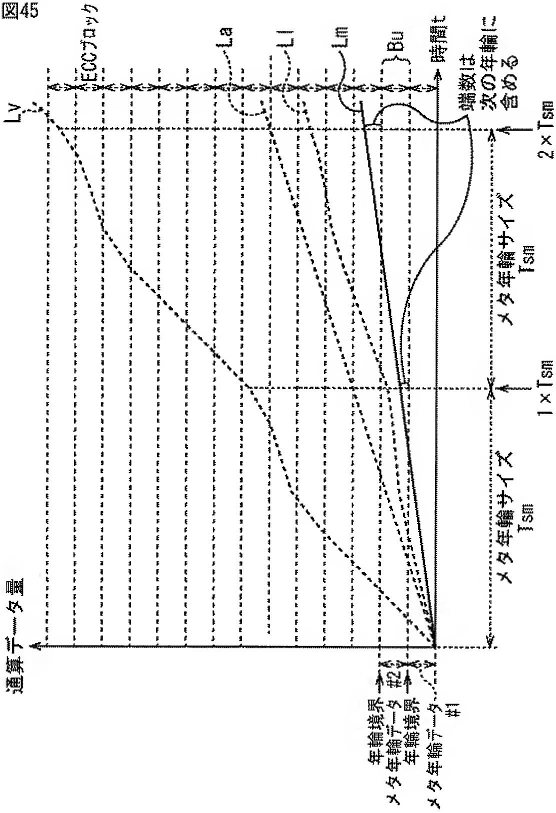
図43


通算データ量
時間t
Lv
ECCブロック
端数は次の年輪に含める
La
Ll
Lm
Bu
端数は次の年輪に含める
年輪境界
画像年輪データ #4
年輪境界
画像年輪データ #3
年輪境界
画像年輪データ #2
年輪境界
画像年輪データ #1
画像年輪サイズ Tsv
1×Tsv
画像年輪サイズ Tsv
2×Tsv
画像年輪サイズ Tsv
3×Tsv
画像年輪サイズ Tsv
4×Tsv

【図４４】
図44

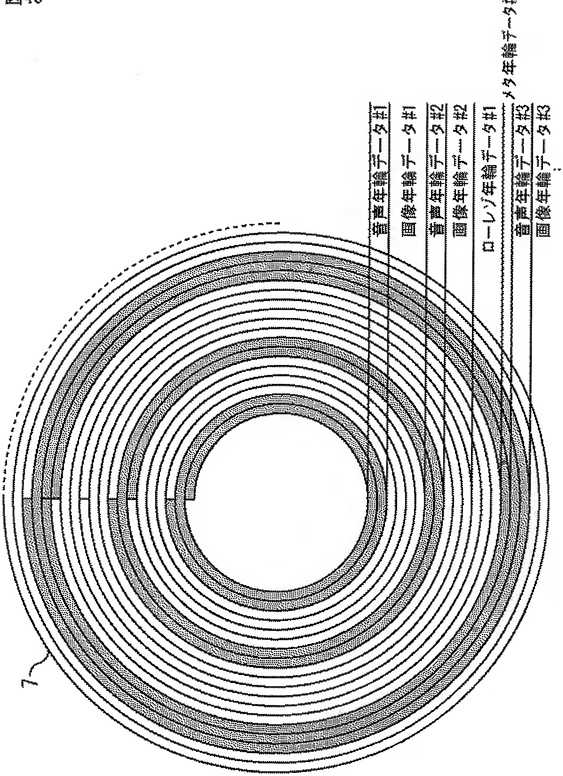

通算データ量
ECCブロック
Ly
La
Ll
Lm
Bu
時間t
端数は次の年輪に含める
2×Ts
1×Ts
ローレゾ年輪サイズ Ts
ローレゾ年輪サイズ Ts
年輪境界
ローレゾ年輪データ #2
年輪境界
ローレゾ年輪データ #1

【図４５】
図45


演算データ量
ECCブロック
Lv
La
Ll
Lm
Bu
時間t
端数は次の年輪に含める
2×Tsm
1×Tsm
メタ年輪サイズ
Tsm
メタ年輪サイズ
Tsm
年輪境界
メタ年輪データ
#2
年輪境界
メタ年輪データ
#1

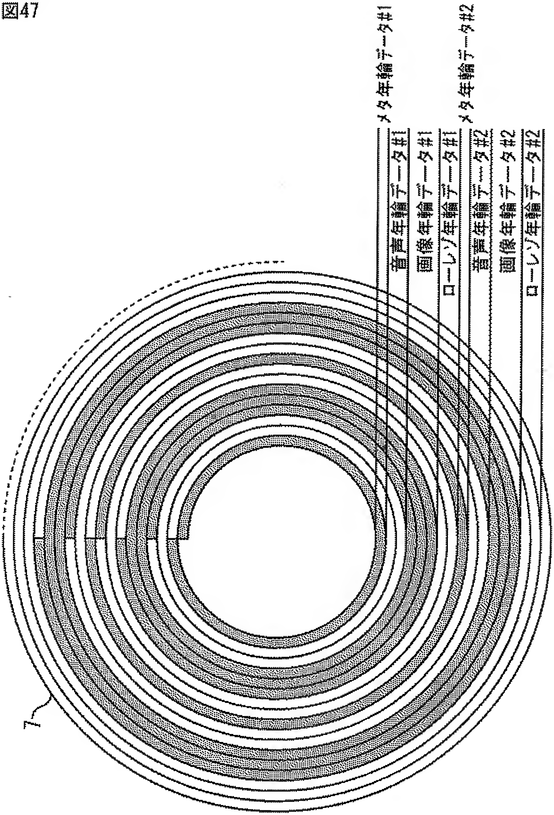
【図４６】

図46

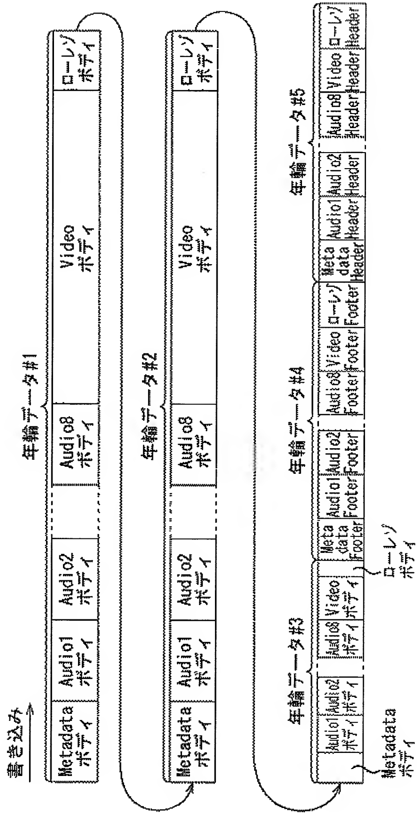

音声年輪データ#1
画像年輪データ#1
音声年輪データ#2
画像年輪データ#2
ローレゾ年輪データ#1
メタ年輪データ#1
音声年輪データ#3
画像年輪データ#3
7

【図４７】

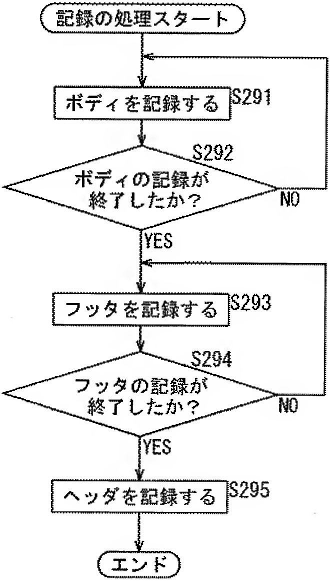
図47


メタ年輪データ#1
音声年輪データ#1
画像年輪データ#1
ローレゾ年輪データ#1
メタ年輪データ#2
音声年輪データ#2
画像年輪データ#2
ローレゾ年輪データ#2
7

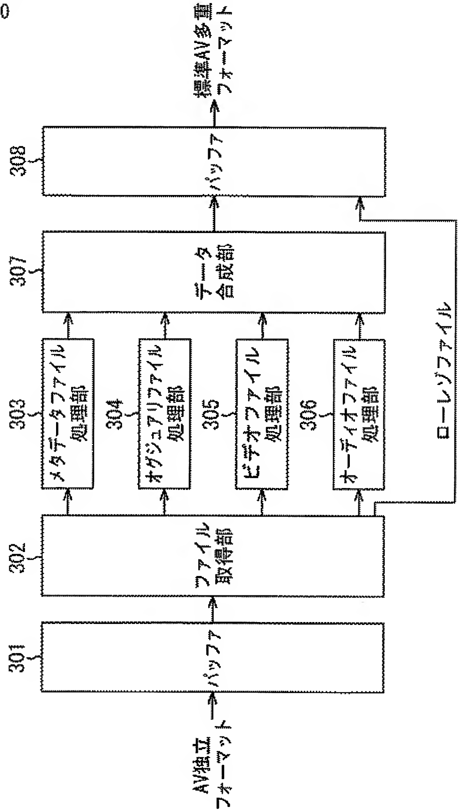
【図４８】
図48


書き込み
Metadata ボディ
Audio1 ボディ
Audio2 ボディ
Audio8 ボディ
Video ボディ
ローレゾ ボディ
年輪データ#1
Metadata ボディ
Audio1 ボディ
Audio2 ボディ
Audio8 ボディ
Video ボディ
ローレゾ ボディ
年輪データ#2
年輪データ#3
Metadata ボディ
Audio1 ボディ
Audio2 ボディ
Audio8 ボディ
Video ボディ
ローレゾ ボディ
年輪データ#4
Meta data Footer
Audio1 Footer
Audio2 Footer
Audio8 Footer
Video Footer
ローレゾ Footer
年輪データ#5
Meta data Header
Audio1 Header
Audio2 Header
Audio8 Header
Video Header
ローレゾ Header

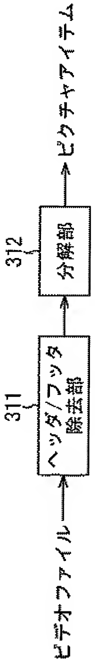
【図４９】
図49


記録の処理スタート
ボディを記録する
S291
S292
ボディの記録が
終了したか？
NO
YES
フッタを記録する
S293
S294
フッタの記録が
終了したか？
NO
YES
ヘッダを記録する
S295
エンド

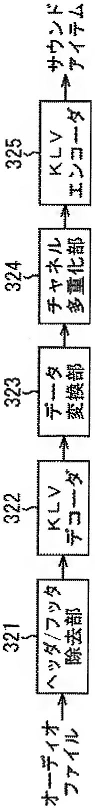
【図５０】
図50


AV独立フォーマット
301
バッファ
302
ファイル取得部
303
メタデータファイル処理部
304
オグジュアリファイル処理部
305
ビデオファイル処理部
306
オーディオファイル処理部
ローレゾファイル
307
データ合成部
308
バッファ
標準AV多重フォーマット

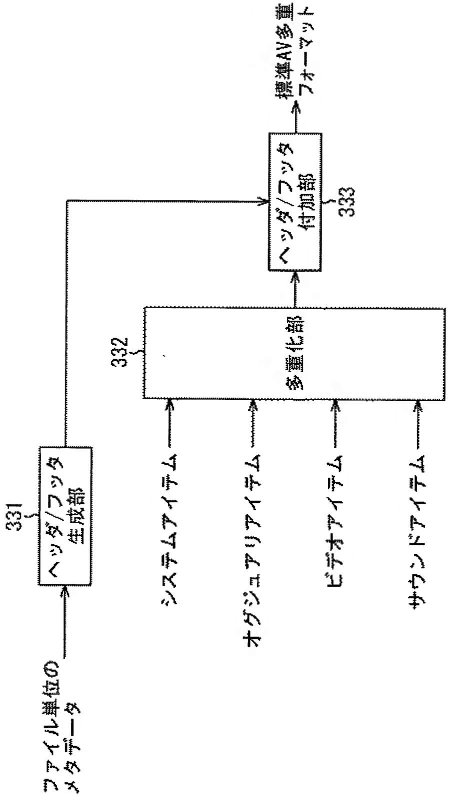
【図５１】
図51



【図５２】
図52

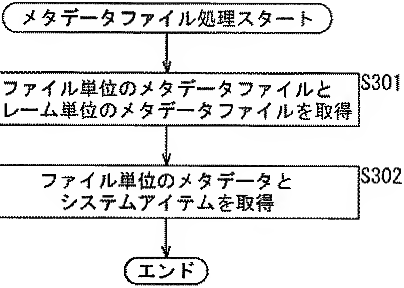

オーディオファイル
321
ヘッダ/フッタ除去部
322
KLV デコーダ
323
データ変換部
324
チャネル多重化部
325
KLV エンコーダ
サウンドアイテム

【図５３】
図53


ファイル単位のメタデータ
ヘッダ/フッタ生成部
331
システムアイテム
オグジュアリアイテム
ビデオアイテム
サウンドアイテム
多重化部
332
ヘッダ/フッタ付加部
333
標準AV多重フォーマット

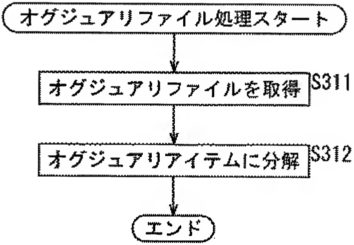
【図５４】

図54


メタデータファイル処理スタート
ファイル単位のメタデータファイルと
フレーム単位のメタデータファイルを取得
S301
ファイル単位のメタデータと
システムアイテムを取得
S302
エンド

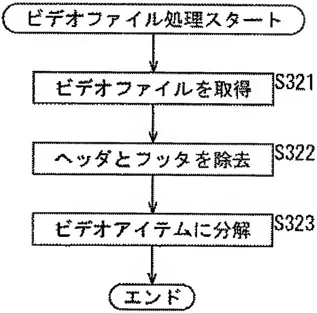
【図５５】

図55

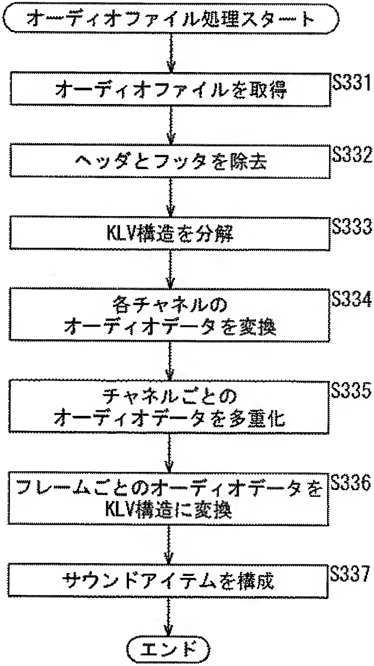

オグジュアリファイル処理スタート
オグジュアリファイルを取得
S311
オグジュアリアイテムに分解
S312
エンド

【図５６】

図56


ビデオファイル処理スタート
ビデオファイルを取得
S321
ヘッダとフッタを除去
S322
ビデオアイテムに分解
S323
エンド

【図５７】
図57


オーディオファイル処理スタート
オーディオファイルを取得
S331
ヘッダとフッタを除去
S332
KLV構造を分解
S333
各チャネルの
オーディオデータを変換
S334
チャネルごとの
オーディオデータを多重化
S335
フレームごとのオーディオデータを
KLV構造に変換
S336
サウンドアイテムを構成
S337
エンド

【図５８】

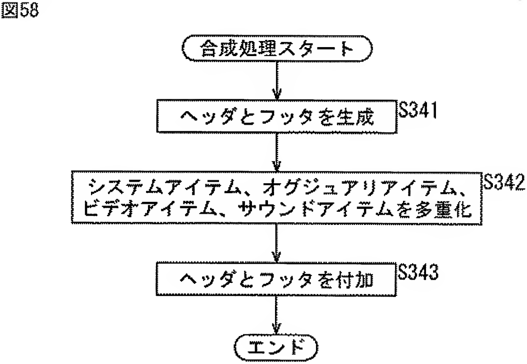
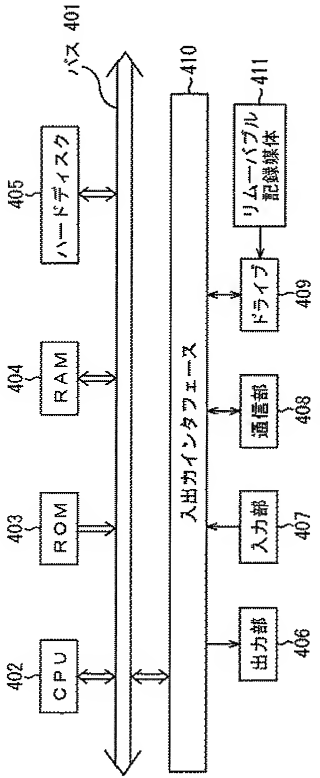


図58
合成処理スタート
ヘッダとフッタを生成
S341
システムアイテム、オグジュアリアイテム、
ビデオアイテム、サウンドアイテムを多重化
S342
ヘッダとフッタを付加
S343
エンド

【図５９】
図59


バス 401
CPU
402
ROM
403
RAM
404
ハードディスク
405
入出力インタフェース
410
出力部
406
入力部
407
通信部
408
ドライブ
409
リムーバブル記録媒体
411

【書類名】図面 [DOCUMENT NAME] DRAWINGS

【図１】 FIGURE 1

１，２	Disk apparatus
３	Editing apparatus
４	Network
５，６	AV apparatus
７	Optical disk
１１	Disk drive apparatus
１２	Format conversion portion
１３	Communication I/F

【図２】 FIGURE 2

【図３】 FIGURE 3

ボディ	Body
ヘッダ	Header
ピクチャアイテム	Picture item
フッタ	Footer
ヘッダ	Header
ボディ	Body
フッタ	Footer
File 単位 Metadata	File based metadata

【図４】 FIGURE 4

ボディ	Body
ヘッダ	Header
ピクチャアイテム	Picture item
フッタ	Footer
セクタ境界	Sector boundary
ECC ブロック境界	ECC block boundary

【図５】 FIGURE 5

ECCブロック境界	ECC block boundary
セクタ境界	Sector boundary

【図６】 FIGURE 6

ECCブロック境界	ECC block boundary

【図７】 FIGURE 7

１２	Format conversion portion
２１	Standard/independent format conversion portion
２２	Independent/standard format conversion portion
AV独立フォーマット	AV independent format
標準AV多重フォーマット	Standard AV multiplexing format

【図８】 FIGURE 8

３１	Buffer
３２	Master file generation portion
３３	Header acquisition portion
３４	Body acquisition portion
３５	Header metadata extraction portion
３６	System item process portion
３７	Metadata file generation portion
３８	Auxiliary item extraction portion
３９	Auxiliary file generation portion
４０	Picture item extraction portion
４１	Video file generation portion
４２	Sound item extraction portion
４３	Audio file generation portion
４４	Buffer
標準AV多重フォーマット	Standard AV multiplexing format
AV独立フォーマット	AV independent format

【図９】	FIGURE 9
５１	Connection portion
５２	Footer generation portion
５３	Header generation portion
５４	Filler generation portion
５５	KLV encoder
ピクチャアイテム	Picture item
ビデオファイル	Video file

【図１０】	FIGURE 10
６１	KLV decoder
６２	Channel separation portion
６３	Data conversion portion
６４	KLV encoder
６５	Header generation portion
６６	Footer generation portion
６７	Filler generation portion
６８	KLV encoder
サウンドアイテム	Sound item
オーディオファイル	Audio file

【図１１】	FIGURE 11
Ｓ１	Generate file names of metadata file, auxiliary file, video file, and audio file
Ｓ２	Generate master file in which XML coded links to file names are placed
マスタファイル生成処理スタート	Start master file generation process
エンド	End

【図１２】	FIGURE 12

Ｓ１１　Obtain header

Ｓ１２　Extract header metadata placed in header

Ｓ１３　Generate file-based metadata file in which header metadata is placed

メタデータファイル生成処理スタート（ファイル単位のメタデータファイル生成処理)　Start metadata file generation process (File-based metadata file generation process)

エンド　End

【図１３】　FIGURE 13

Ｓ２１　Obtain body

Ｓ２２　Extract system item

Ｓ２３　Add filler to each system item

Ｓ２４　Connect system item

Ｓ２５　Output body

Ｓ２６　Generate footer

Ｓ２７　Generate filler of footer

Ｓ２８　Output footer

Ｓ２９　Generate header

Ｓ３０　Generate filler of header

Ｓ３１　Output header

メタデータファイル生成処理スタート（フレーム単位のメタデータファイル生成処理)　Start metadata file generation process (Flame-based metadata file generation process)

エンド　End

【図１４】　FIGURE 14

Ｓ４１　Obtain body

Ｓ４２　Extract auxiliary item

Ｓ４３　Connect all auxiliary items to generate auxiliary file

オグジュアリファイル生成処理スタート

エンド	End
【図１５】	FIGURE 15
Ｓ５１	Obtain body
Ｓ５２	Extract picture item
Ｓ５３	Connect picture items
Ｓ５４	Is picture item last?
Ｓ５５	Output body
Ｓ５６	Generate filler for last picture item
Ｓ５７	Convert filler for last picture item into KLV structure
Ｓ５８	Output body
Ｓ５９	Generate footer
Ｓ６０	Generate filler for footer
Ｓ６１	Output footer
Ｓ６２	Generate header
Ｓ６３	Generate filler for header
Ｓ６４	Output header
ビデオファイル生成処理スタート	Start video file generation process
エンド	End
【図１６】	FIGURE 16
Ｓ７１	Obtain body
Ｓ７２	Extract audio item
Ｓ７３	Decompose KLV structure
Ｓ７４	Separate channel-based audio data
Ｓ７５	Convert audio data for each channel
Ｓ７６	Convert audio data for each channel into KLV structure
Ｓ７７	Generate filler for each channel's body
Ｓ７８	Convert filler for each channel's body into KLV structure
Ｓ７９	Output body for each channel

Ｓ８０	Generate footer for each channel
Ｓ８１	Generate filler for each channel's footer
Ｓ８２	Output footer for each channel
Ｓ８３	Generate header for each channel
Ｓ８４	Generate filler for each channel's header
Ｓ８５	Output header for each channel
オーディオファイル生成処理スタート	Start audio file generation process
エンド	End

【図１７】 FIGURE 17

１２	Format conversion portion
１１１	Spindle motor
１１２	Pickup portion
１１３	RF amplifier
１１４	Servo control portion
１１５	Signal process portion
１１６	Memory controller
１１７	Memory
１１８	Data conversion portion
１１９	Control portion
１２０	Operation portion

【図１８】 FIGURE 18

１４１	Data amount detection portion
１４２	Low resolution data generation portion
ビデオファイル	Video file
オーディオファイル	Audio file
メタデータファイル	Metadata file
フォーマット変換部１２から	From format conversion portion 12
メモリコントローラ１１６へ	To memory controller 116
メモリコントローラ１１６より	From memory controller 116

ローレゾデータファイル	Low resolution data file
フォーマット変換部１２へ	To format conversion portion 12

【図１９】　FIGURE 19

ECC ブロック境界	ECC block boundary

【図２０】　FIGURE 20

ECC ブロックサイズ	ECC block size
ECC ブロック境界	ECC block boundary

【図２１】　FIGURE 21

【図２２】　FIGURE 22

【図２３】　FIGURE 23

【図２４】　FIGURE 24

【図２５】　FIGURE 25

１６１	Buffer
１６２	File process portion
１６３	Metadata file process portion
１６４	Video file process portion
１６５	Audio file process portion
１６６	Data synthesis portion
１６７	Buffer
ローレゾファイル	Low resolution file
メタデータファイル	Metadata file
ビデオファイル	Video file
オーディファイル	Audio file

【図２６】　FIGURE 26

１８１	Decomposition portion
１８２	Data conversion portion
１８３	KLV encoder
ピクチャエッセンス	Picture essence
ビデオファイル	Video file

【図２７】　FIGURE 27

２０１	KLV decoder
２０２	Data conversion portion
２０３	Channel multiplexing portion
２０４	KLV encoder
２０５	Filler generation portion
２０６	KLV encoder
サウンドエッセンス	Sound essence
オーディオファイル	Audio file

【図２８】　FIGURE 28

２２１	Multiplexing portion
２２２	Footer generation portion
２２３	Header generation portion
２２４	Filler generation portion
ローレゾファイル	Low resolution file
システムアイテム	System item
ビデオエッセンス	Video essence
サウンドエッセンス	Sound essence

【図２９】　FIGURE 29

Ｓ１０１	Obtain video file body
Ｓ１０２	Decompose video file into picture item

Ｓ１０３	Convert picture item
Ｓ１０４	Convert picture item into KLV
structure	
ビデオファイル処理スタート	Start video file process
エンド	End

【図３０】 FIGURE 30

Ｓ１２１	Obtain audio file body
Ｓ１２２	Decompose KLV structure
Ｓ１２３	Convert audio data
Ｓ１２４	Multiplex audio data
Ｓ１２５	Convert audio data into KLV structure
Ｓ１２６	Generate filler
Ｓ１２７	Convert filler to KLV structure
オーディオファイル処理スタート	Start audio file process
エンド	End

【図３１】 FIGURE 31

Ｓ１４１	Obtain metadata file body
Ｓ１４２	Generate filler
Ｓ１４３	Convert filler into KLV structure
メタデータファイル処理スタート	Start metadata file process
エンド	End

【図３２】 FIGURE 32

Ｓ１６１	Multiplex system essence, video
essence, and audio essence	
Ｓ１６２	Add body partition
Ｓ１６３	Output body
Ｓ１６４	Generate footer
Ｓ１６５	Output footer

Ｓ１６６	Generate header
Ｓ１６７	Generate filler
Ｓ１６８	Output header
ローレゾファイル合成の処理スタート	Start low resolution file synthesis process
エンド	End

【図３３】 FIGURE 33

Ｓ２３１	Set audio tree ring size Tsa, video tree ring size Tsv, Low resolution tree ring size Tsl, meta tree ring size Tsm
Ｓ２３２	Start low resolution data generation process, audio data storage process, video data storage process, metadata storage process, and low resolution data storage process
Ｓ２３３	Start audio data storage task
Ｓ２３４	Start video data storage task
Ｓ２３５	Start low resolution data storage task
Ｓ２３６	Start metadata storage task
Ｓ２３７	Is there instruction to terminate recording?
Ｓ２３８	Have all recording tasks terminated?
Ｓ２３９	Terminate low resolution data generation process, audio data storage process, video data storage process, metadata storage process, low resolution data storage process
Ｓ２４０	Have all storage processes terminated?
記録処理スタート	Start storage process
エンド	End

【図３４】 FIGURE 34

Ｓ２５３	Is audio data supplied?
Ｓ２５４	Has audio data of audio signal equivalent to Tsa x Na been stored in memory?
Ｓ２５５	Read to extract audio data equivalent to integral

multiple to ECC block from audio data stored in memory

Ｓ２５６ Supply audio data to signal processing portion to control recording so as to record audio data equivalent to integral multiple of ECC block in as many ECC blocks as integral multiple

Ｓ２５８ Read audio data remaining in memory to control recording so as to record audio data equivalent to integral multiple of ECC block in as many ECC blocks as integral multiple

音声データ記録タスクスタート Start audio data recording task

エンド End

【図３５】 FIGURE 35

通算データ量 Total data amount

時間 t Time t

【図３６】 FIGURE 36

ECC ブロック ECC block

【図３７】 FIGURE 37

Ｓ２６３ Is video data supplied?

Ｓ２６４ Has video data of video signal equivalent to Tsa x Na been stored in memory?

Ｓ２６５ Read to extract video data equivalent to integral multiple to ECC block from video data stored in memory

Ｓ２６６ Supply video data to signal processing portion to control recording so as to record video data equivalent to integral multiple of ECC block in as many ECC blocks as integral multiple

Ｓ２６８ Read video data remaining in memory to control recording so as to record video data equivalent to integral multiple of ECC block in as many ECC blocks as integral multiple

画像データ記録タスクスタート Start video data recording task

エンド

【図３８】 FIGURE 38

通算データ量	Total data amount
時間ｔ	Time t

【図３９】 FIGURE 39

Ｓ２７３	Is low resolution data supplied?
Ｓ２７４	Has low resolution data of low resolution signal equivalent to Tsa x Na been stored in memory?
Ｓ２７５	Read to extract low resolution data equivalent to integral multiple to ECC block from low resolution data stored in memory
Ｓ２７６	Supply low resolution data to signal processing portion to control recording so as to record low resolution data equivalent to integral multiple of ECC block in as many ECC blocks as integral multiple
Ｓ２７８	Read low resolution data remaining in memory to control recording so as to record low resolution data equivalent to integral multiple of ECC block in as many ECC blocks as integral multiple
ローレゾデータ記録タスクスタート	Start low resolution data recording task
エンド	

【図４０】 FIGURE 40

Ｓ２８３	Is metadata supplied?
Ｓ２８４	Has metadata of low resolution signal equivalent to Tsa x Na been stored in memory?
Ｓ２８５	Read to extract metadata equivalent to integral multiple to ECC block from metadata stored in memory
Ｓ２８６	Supply metadata to signal processing portion to control recording so as to record metadata equivalent to integral multiple of ECC block in as many ECC blocks as integral multiple

Ｓ２８８	Read metadata remaining in memory to control recording so as to record metadata equivalent to integral multiple of ECC block in as many ECC blocks as integral multiple
メタデータ記録タスクスタート	Start metadata recording task
エンド	

【図４１】 FIGURE 41

画像データの通算データ量	Total data amount of video data
通算データ量	Total data amount
ECC ブロック	ECC block
ローレゾデータの通算データ量	Total data amount of low resolution data
メタデータの通算データ量	Total data amount of metadata
時間t	Time t

【図４２】 FIGURE 42

通算データ量	Total data amount
ECC ブロック	ECC block
年輪境界	Tree ring boundary
音声年輪データ	audio tree ring data
時間t	Time T
端数は次の年輪に含める	Fractions are included into next tree ring
音声年輪サイズ	Audio tree ting size

【図４３】 FIGURE 43

通算データ量	Total data amount
年輪境界	Tree ring boundary
画像年輪データ	Video tree ring data
ECC ブロック	ECC block

端数は次の年輪に含める	Fractions are included into next tree ring
時間 t	Time t
画像年輪サイズ	Video tree ring size

【図４４】　FIGURE 44

通算データ量	Total data amount
ECC ブロック	ECC blocks
年輪境界	Tree ring boundary
ローレゾ年輪データ	Low resolution tree ring data
時間 t	Time t
端数は次の年輪に含める	Fractions are included into next tree ring
ローレゾ年輪サイズ	Low resolution tree ring size

【図４５】　FIGURE 45

通算データ量	Total data amount
ECC ブロック	ECC blocks
年輪境界	Tree ring boundary
メタ年輪データ	Meta tree ring data
時間 t	Time t
端数は次の年輪に含める	Fractions are included into next tree ring
メタ年輪サイズ	Meta tree ring size

【図４６】　FIGURE 46

音声年輪データ	Audio tree ring data
画像年輪データ	Video tree ring data
ローレゾ年輪データ	Low resolution tree ring data
メタ年輪データ	Meta tree ring data

【図４７】 FIGURE 47

メタ年輪データ	Meta tree ring data
音声年輪データ	Audio tree ring data
画像年輪データ	Video tree ring data
ローレゾ年輪データ	Low resolution tree ring data

【図４８】 FIGURE 48

書き込み	Write
年輪データ	Tree ring data
ボディ	Body
ローレゾボディ	Low resolution body
ローレゾ Footer	Low resolution footer

【図４９】 FIGURE 49

Ｓ２９１	Record body
Ｓ２９２	Has body recording terminated?
Ｓ２９３	Record footer
Ｓ２９４	Record header
記録の処理スタート	Start recording process
エンド	End

【図５０】 FIGURE 50

３０１	Buffer
３０２	File acquisition portion
３０３	Metadata file process portion
３０４	Auxiliary file process portion
３０５	Video file process portion
３０６	Audio file process portion
３０７	Data synthesis portion
３０８	Buffer
AV 独立フォーマット	AV independent format

標準AV多重フォーマット	Standard AV multiplexing format
ローレゾファイル	Low resolution file

【図５１】 FIGURE 51

３１１	Header/footer removal portion
３１２	Decomposition portion
ビデオファイル	Video file
ピクチャアイテム	Picture item

【図５２】 FIGURE 52

３２１	Header/footer removal portion
３２２	KLV decoder
３２３	Data conversion portion
３２４	Channel multiplexing portion
３２５	KLV encoder
オーディオファイル	Audio file
サウンドアイテム	Sound item

【図５３】 FIGURE 53

３３１	Header/ footer generation portion
３３２	Multiplexing portion
３３３	Header/footer adding portion
ファイル単位のメタデータ	File-based metadata
システムアイテム	System item
オグジュアリアイテム	Auxiliary item
ビデオアイテム	Video item
サウンドアイテム	Sound item
標準AV多重フォーマット	Standard AV multiplexing format

【図５４】 FIGURE 54

Ｓ３０１	Obtain file-based metadata file and

	frame-based metadata file
Ｓ３０２	Obtain file-based metadata and system item
メタデータファイル処理スタート	Start metadata file process
エンド	End

【図５５】 FIGURE 55

Ｓ３１１	Obtain auxiliary file
Ｓ３１２	Decompose auxiliary file into auxiliary items
オグジュアリファイル処理スタート	Start auxiliary file process
エンド	End

【図５６】 FIGURE 56

Ｓ３２１	Obtain video file
Ｓ３２２	Remove header and footer
Ｓ３２３	Decompose video file into video items
ビデオファイル処理スタート	Start video file process
エンド	End

【図５７】

Ｓ３３１	Obtain audio file
Ｓ３３２	Remove header and footer
Ｓ３３３	Decompose KLV structure
Ｓ３３４	Convert audio data for each channel
Ｓ３３５	Multiplex channel-based audio data
Ｓ３３６	Convert frame-based audio data into KLV structure
Ｓ３３７	Configure sound item
オーディオファイル処理スタート	Start audio file process

エンド	End
【図５８】	
Ｓ３４１	Generate header and footer
Ｓ３４２	Multiplex system item, auxiliary item, video item, and sound item
Ｓ３４３	Add header and footer
合成処理スタート	Start synthesis process
エンド	End
【図５９】	
４０１	Bus
４０５	Hard disk
４０６	Output portion
４０７	Input portion
４０８	Communication portion
４０９	Drive
４１０	Input/output interface
４１１	Removable recording medium

[NAME OF DOCUMENT] ABSTRACT

[SUMMARY]

[OBJECT]

To improve the usability of recording media and achieve more efficient read and write processes.

[MEANS FOR SOLVING]

A header generation portion 53 generates a header to be arranged at the beginning of a file. A footer generation portion 52 generated a footer to be arranged at the last of the file. A filler generation portion 54 generates a filler which allows the data amount of each of a body, the header, or the footer to be an integral fraction of an ECC block of an optical disk by adding the filler to the body, the header, or the footer. A KLV encoder 55 KLV encodes the filler to be added to the body in accordance with a KLV structure. The present invention can be applied to a disk apparatus that records video data of audio data on an optical disk.

[SELECTED DRAWING] Figure 9